# LĪGUMS

***Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrādi***

*(iepirkuma identifikācijas numurs NVA 2018/19)*

Rīgā

2018.gada 15.augusts Nr. 1.1-34/21

**Nodarbinātības valsts aģentūra**, tās direktora pienākumu izpildītājas Kristīnes Stašānes personā, kura rīkojas saskaņā ar Labklājības ministrijas 2018.gada 18.jūlija rīkojumu Nr.23.1-1-02/36 „Par ikgadējā atvaļinājuma piešķiršanu " (turpmāk – Pasūtītājs), no vienas puses, un

**SIA Uniso**, reģ.Nr. 40003562863, tās Valdes locekļa Mārtiņa Rumkovska personā, kurš darbojas uz statūtu pamata (turpmāk – Izpildītājs), no otras puses,

turpmāk abi kopā saukti Puses, bet katrs atsevišķi – Puse,

pamatojoties uz iepirkuma "Bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrāde" (iepirkuma identifikācijas Nr. *NVA 2018/19* (turpmāk – atklāts konkurss) rezultātiem,

noslēdz šo līgumu (turpmāk – Līgums):

## 1. LĪGUMA PRIEKŠMETS

1.1. Pasūtītājs pasūta un Izpildītājs apņemas saskaņā ar Tehnisko specifikāciju (1.pielikums) veikt bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas (turpmāk – "BURVIS") papildināšanu ar moduli informācijas apmaiņai ar Eiropas darba mobilitātes portālu EURES (turpmāk – "EURES modulis", arī "Sistēma"), turpmāk – Pakalpojumi.

## 2. LĪGUMA SUMMA

2.1. Līgumcena par Izpildītāja sniegtajiem Pakalpojumiem ir *EUR 41900* (*Četrdesmit viens tūkstotis deviņi simti euro*) bez pievienotās vērtības nodokļa (PVN). Pievienotās vērtības nodoklis tiek pieskaitīts līgumcenai saskaņā ar spēkā esošajiem normatīvajiem aktiem.

2.2. Līgumcena ietver visas izmaksas, kuras nepieciešamas Tehniskajā specifikācijā norādīto darbu veikšanai, testēšanai un atkļūdošanai un ieviešanai ekspluatācijā.

## 3. IZSTRĀDES DARBU VEIKŠANAS KĀRTĪBA.

3.1. Pakalpojums Izpildītājam jāsniedz līdz 2018.gada 17.decembrim. Ar pakalpojuma sniegšanas termiņu saprotams termiņš, kurā visi tehniskajā specifikācijā norādītie darbi ir pilnībā pabeigti, EURES modulis ir uzstādīts produkcijas vidē un gatavs ekspluatācijai, par ko ir parakstīts darbu nodošanas – pieņemšanas akts.

3.2. Pakalpojums sniedzams secīgos posmos, atbilstoši funkcionālām sadaļām:

3.2.1. 1.posms - reģistrēto vakanču datu apmaiņas tīmekļa saskarnes izveidošana un

testēšana saņemot EK apstiprinājumu par tehnisko uzdevumu izpildi testa vidē;

3.2.2. 2.posms - darba meklētāju CV datu apmaiņas tīmekļa saskarnes izveidošana un testēšana saņemot EK apstiprinājumu par tehnisko uzdevumu izpildi testa vidē;

3.2.3. 3.posms - izstrādāto saskarņu izvietošana produkcijas vidē, saņemot EK apstiprinājumu par korektu produkcijas datu apmaiņas uzsākšanu.

3.3. Pakalpojuma akceptēšanas kritēriji:

3.3.1. Katra funkcionālā sadaļa tiek akceptēta, ja nodevuma sastāvs atbilst Līguma 1.pielikuma prasībām, izstrādāts un iesniegts funkcionālās sadaļas projektējuma apraksts, dokumentēts pirmkods un kompilēts kods (izpildkods), veikta akcepttestēšana, tajā skaitā integrācijas testēšana ar EURES portālu (https://ec.europa.eu/eures/public/lv/homepage) un akcepttesta laikā nav konstatēta neviena 1.- 3. kategorijas problēma un ne vairāk, kā 5 (piecas) 4.kategorijas problēmas. Konstatētām drošības problēmām, atkarībā no to ietekmes, tiek piešķirta 1.kategorija vai 2.kategorija.

3.3.2. EURES modulis/funkcionālā sadaļa nedrīkst radīt BURVIS darbības traucējumus vai drošības apdraudējumus. Šādos gadījumos darbi netiks pieņemti līdz BURVIS darbības traucējumu vai drošības apdraudējumu novēršanai.

3.3.3. Puses var vienoties par EURES moduļa ekspluatācijas uzsākšanu un akceptēšanu arī ar noteiktām 3.kategorijas problēmām vai ar 4.kategorijas problēmām, savstarpēji vienojoties par katras šādas problēmas novēršanas termiņu garantijas saistību ietvaros (bezatlīdzības izpilde).

3.3.4. EURES modulis tiek uzskatīts par akceptētu, ja saskaņā ar Pasūtītāja rīkojumu faktiski tiek uzsākta tā lietošana produkcijas vidē un Pasūtītājam nav radušies iebildumi pret EURES moduļa darbību. Tādā gadījumā tiek parakstīts pieņemšanas nodošanas akts ar attiecīgu atrunu.

3.4. Testēšanu Pasūtītājs ir tiesīgs veikt 10 (desmit) darba dienu laikā. Gadījumā, ja testēšanai nepieciešams ilgāks laiks, testēšanai nepieciešamais papildus laiks netiek ieskaitīts Pakalpojuma sniegšanas termiņā.

3.5. Gadījumā, ja testēšanas laikā tiek atklātas 1.-3.kategorijas problēmas, kuras Izpildītājs nevar operatīvi (ne vairāk, kā 2 (divu) darba dienu laikā) novērst, testēšana tiek uzskatīta par neveiksmīgu un Izpildītājam par saviem līdzekļiem jānovērš EURES moduļa / funkcionālās sadaļas defekti un jāiesniedz Pasūtītājam labota programmatūra atkārtotai testēšanai.

3.6. Par darbu izpildes termiņu tiek uzskatīts datums, kurā parakstīts darbu nodošanas – pieņemšanas akts.

## 4. NORĒĶINU KĀRTĪBA

4.1. Pēc pirmā posma darbu nodošanas – pieņemšanas akta parakstīšanas Izpildītājs izraksta rēķinu par 35% (trīsdesmit pieciem procentiem) no Līguma 2.1.punktā norādītās

līgumcenas.

4.2. Pēc otrā posma darbu nodošanas – pieņemšanas akta parakstīšanas Izpildītājs izraksta rēķinu par 35% (trīsdesmit pieciem procentiem) no Līguma 2.1.punktā norādītās līgumcenas.

4.3. Pēc trešā posma darbu nodošanas – pieņemšanas akta parakstīšanas Izpildītājs izraksta rēķinu par 30% (trīsdesmit procentiem) no Līguma 2.1.punktā norādītās darbu līgumcenas.

4.4. Pasūtītājs apmaksā visus uz Līguma pamata pamatoti izrakstītos rēķinus 30 (trīsdesmit) dienu laikā no to saņemšanas brīža.

## 5. LĪGUMA PĀRVALDĪBA UN ATBILDĪGĀS PERSONAS

5.1. Līguma izpildei katra no Pusēm nozīmē savu pārstāvi, kuru pienākums ir vadīt un sekot Līguma izpildei un informēt par Līguma izpildi gan savu, gan arī otru līgumslēdzēju Pusi. Jebkura informācija, kura jānosūta otrai līgumslēdzējai pusei, ir uzskatāma par nosūtītu, ja tā nosūtīta otras puses kontaktpersonai un uz oficiālo otras puses e-pasta adresi. Jebkurai informācijai, ja tā rada līgumslēdzējām pusēm saistošas juridiskas sekas, jābūt nosūtītai elektroniski parakstīta dokumenta veidā.

**5.2.** Pasūtītāja nozīmētais pārstāvis: ISUAN (Informācijas sistēmu uzturēšanas un attīstības nodaļa) vadītājs Raitis Berkmanis, e – pasts: raitis.berkmanis@nva.gov.lv. Pasūtītāja pārstāvis šī līguma ietvaros ir tiesīgs koordinēt darbus, pieteikt pretenzijas, parakstīt darbu nodošanas – pieņemšanas aktus.

5.3. Izpildītāja nozīmētie pārstāvji: Mārtiņš Rumkovskis, e – pasts: martins@uniso.lv Izpildītāja pārstāvis šī līguma ietvaros ir tiesīgs koordinēt darbus, pieteikt pretenzijas, parakstīt darbu nodošanas – pieņemšanas aktus.

5.4. Puse savu pārstāvju nomaiņas gadījumā otru Pusi par to rakstiski informē 3 (trīs) darba dienu laikā.

## 6. PUŠU PIENĀKUMI UN TIESĪBS

6.1. Izpildītājs apņemas:

6.1.1. ievērot Līguma nosacījumus;

6.1.2. veikt Pakalpojumu izpildi saskaņā ar Līguma pielikumos noteikto, ņemot vērā starp Pusēm noslēgtās atsevišķās Vienošanās;

6.1.3. nodrošināt, ka Pakalpojuma sniegšanu veiks vadošie speciālisti, kuri norādīti Līguma 3.pielikumā. Gadījumā, ja Līguma 3.pielikumā norādīto speciālistu personisku piedalīšanos nav iespējams nodrošināt, Izpildītājam ir pienākums nodrošināt līdzvērtīgas kvalifikācijas speciālistu iesaisti Līguma izpildē, saskaņojot to ar Pasūtītāju Publisko iepirkumu likuma 62.panta prasībām;

6.1.4. izmantojot Pasūtītāja tehnisko un sistēmu nodrošinājumu, ievērot Pasūtītāja noteikumus, Pasūtītāja Informācijas drošības politiku un prasības informācijas drošības jomā;

6.1.5. darbu izpildē sadarboties ar Pasūtītāja norādītajiem BURVIS uzturēšanas pakalpojuma sniedzējiem, kā arī nepieciešamības gadījumā patstāvīgi komunicēt ar EURES portāla tehniskā atbalsta struktūrvienību (komunikācija notiek angļu valodā).

6.2. Pasūtītājs apņemas:

6.2.1. ievērot Līguma nosacījumus;

6.2.2. Līguma izpildes ietvaros koordinēt sadarbību ar BURVIS uzturēšanas pakalpojuma sniedzēju;

6.2.3. Ja Pakalpojuma sniegšanai nepieciešamas BURVIS izmaiņas, pasūtīt BURVIS uzturētājam šādu izmaiņu realizāciju, ja tā ir objektīvs priekšnoteikums Pakalpojuma izpildei;

6.2.4. sniegt Izpildītājam visu nepieciešamo informāciju, kas nepieciešama Līguma izpildes ietvaros;

6.2.5. pieņemt Izpildītāja kvalitatīvi sniegtos Pakalpojumus, kas atbilst Līgumā noteiktajām prasībām, un veikt samaksu saskaņā ar Līguma nosacījumiem;

6.2.6. nodrošināt Izpildītāja personālam iespēju izmantot Pasūtītāja tehnisko un sistēmu nodrošinājumu, ja tas nepieciešams Izpildītājam Līguma saistību veikšanai un ir saskaņots ar Pasūtītāju.

6.3. Ja Izpildītājs nepilda Līgumā paredzētos pienākumus, Pasūtītājs pēc to konstatēšanas ne vēlāk kā 5 (piecu) darba dienu laikā rakstiski par to informē Izpildītāju.

6.4. Ja Pasūtītājs nepilda Līgumā paredzētos pienākumus, Izpildītājs pēc to konstatēšanas ne vēlāk kā 5 (piecu) darba dienu laikā rakstiski par to informē Pasūtītāju.

6.5. Puses apņemas nekavējoties rakstiski informēt viena otru par jebkādām grūtībām Līguma izpildes procesā, kas varētu aizkavēt savlaicīgu Pakalpojumu sniegšanu un Līguma izpildi.

6.6. Ja vienas Puses saistību izpildes nokavējums (tikai tāds nokavējums, kas ietekmē otras Puses spējas izpildīt savas saistības) liedz otrai Pusei veikt savlaicīgu saistību izpildi, otras Puses saistību izpildes termiņš tiek pagarināts par pirmās Puses nokavēto laika posmu. Pusei, kura prasa, lai minēto apstākļu dēļ tiktu pagarināts saistību izpildes termiņš, ir pienākums iesniegt pierādījumus, kas pamato otras Puses saistību izpildes nokavējuma faktu.

6.7. Izpildītājam ir tiesības slēgt līgumus ar citām fiziskām vai juridiskām personām, lai izpildītu daļu no veicamā darba apjoma. Šādos gadījumos Izpildītājs ir atbildīgs par apakšuzņēmēja veikumu tādā pašā mērā kā par savējo, kā arī par to, lai apakšuzņēmējs pilnībā ievērotu Līgumā noteiktos konfidencialitātes noteikumus. Gadījumā, ja apakšuzņēmējam nododamo darbu apjoms ir vismaz 10% no Līgumcenas, Izpildītājam apakšuzņēmēja iesaiste vai nomaiņa jāsaskaņo ar Pasūtītāju, saskaņā ar Publisko iepirkumu likuma 62.panta prasībām.

6.8. Pasūtītājam ir tiesības par saviem līdzekļiem veikt kontroli par Līguma izpildi, t.sk.

patstāvīgi vai ar pilnvarotas trešās puses starpniecību veikt Izpildītāja darbības auditu saistībā ar sniegtajiem Pakalpojumiem, pieaicinot speciālistus un ekspertus.

6.9. Pasūtītājam tiesības dot Izpildītājam obligāti izpildāmus norādījumus jautājumos, kas saistīti ar līguma godprātīgu, kvalitatīvu, savlaicīgu un normatīvajiem aktiem atbilstošu izpildi;

## 7. PUŠU ATBILDĪBA

7.1. Par Pakalpojuma sniegšanas termiņa kavējumu Pasūtītājs ir tiesīgs pieprasīt no Izpildītāja līgumsodu 0,5 % (piecas desmitdaļas no procenta) apmērā no līgumcenas par katru kavējuma dienu, bet ne vairāk kā 10 % (desmit procenti) no līgumcenas, ja Puses par to nav vienojušās citādi.

7.2. Ja pēc trīs akcepttestēšanas kārtām Pakalpojums neatbilst Līguma 3.2.punktā noteiktajiem akceptēšanas kritērijiem, Pasūtītājs ir tiesīgs izbeigt Līgumu bez samaksas par Pakalpojumu sniegšanas, kā arī kompensācijas un zaudējumu atlīdzības.

7.3. Ja Izpildītājs nenodrošina līgumā noteiktos reakcijas laikus vai problēmas novēršanas laikus, līgumsodi Izpildītājam tiek aprēķināti šādā apmērā:

7.3.1. 1.prioritātes kļūdas (avārija) gadījumā – 100,00 EUR (viens simts euro) par katru kavēto stundu darba dienās, darba laikā;

7.3.2. 2.prioritātes kļūdas (kritiska kļūda) gadījumā – 50,00 EUR (piecdesmit euro) par katru kavēto stundu darba dienās, darba laikā;

7.3.3. 3.prioritātes kļūdas (būtiska kļūda) gadījumā – 50,00 EUR (piecdesmit euro) par katru kavēto darba dienu;

7.4. Par Līguma 8. punktā noteikto konfidencialitātes noteikumu, tajā skaitā personas datu apstrādes noteikumu pārkāpumu Izpildītājs maksā līgumsodu 1000,00 EUR (viens tūkstotis euro) par katru gadījumu, kur konstatējama Izpildītāja vaina, kā arī sedz visus zaudējumus, kas radušies ar konkrēto nepamatotu datu apstrādes gadījumu.

7.5. Par Līgumā noteikto maksājumu kavējumu Izpildītājs ir tiesīgs pieprasīt no Pasūtītāja līgumsodu 0,5% (piecas desmitdaļas no procenta) apmērā no laikā nesamaksātās summas par katru kavējuma dienu, bet ne vairāk par 10 % (desmit procenti) no attiecīgā Pasūtījuma summas, ja Puses par to nav vienojušās citādi.

7.6. Ja Pasūtītājs vienpusēji izbeidzis līgumu tādas Izpildītāja saistību neizpildes dēļ, kuras dēļ Pasūtītājs nav ieinteresēts turpināt Līgumu (Līguma 7.2.punkts), Izpildītājs maksā Pasūtītājam līgumsodu 10% (desmit procentu) apmērā no Līguma 2.1.punktā noteiktās līgumcenas.

7.7. Izpildītājs nav atbildīgs par tiešajiem vai netiešajiem zaudējumiem, kas Pasūtītājam radušies Sistēmas lietošanas, apmācību vai Sistēmas papildinājumu izstrādes, piegādes vai ieviešanas pakalpojumu sniegšanas rezultātā.

7.8. Izpildītājs ir atbildīgs par tiešajiem vai netiešajiem zaudējumiem Sistēmas papildinājumu izstrādes, piegādes vai ieviešanas pakalpojumu sniegšanas rezultātā, kas ir noticis

Izpildītāja rupjas neuzmanības vai ļauna nolūka dēļ. Šādos gadījumos kopējā Izpildītāja atbildība nedrīkst pārsniegt attiecīgo Pakalpojumu sniegšanas izmaksas.

7.9. Pasūtītājam nav pienākums apmaksāt jebkādus Izpildītāja izdevumus un zaudējumus par tiem Pakalpojumiem, kurus Izpildītājs nav veicis vai par kuriem Līgumā noteiktajā kārtībā ir konstatētas nepilnības, kas no Izpildītāja puses nav novērstas.

7.10. Pasūtītājam ir tiesības vienpusēji izbeigt Līgumu, ja:

7.10.1. ir stājies spēkā tiesas spriedums par Izpildītāja atzīšanu par maksātnespējīgu;

7.10.2. ir notikusi Izpildītāja labprātīga vai piespiedu likvidācija;

7.10.3. Izpildītāja saimnieciskā darbība ir apturēta vai pārtraukta;

7.10.4. pret Izpildītāju tikušas vērstas darbības, kas saistītas ar aresta uzlikšanu vairāk kā 50 % (piecdesmit procenti) no Izpildītāja bilances aktīviem;

7.10.5. iestājas ārēji, no Pasūtītāja neatkarīgi apstākļi (t.sk. Saeimas, Ministru kabineta, Labklājības ministrijas lēmumi, ar kuriem tiek samazināts Pasūtītājam pieejamais budžeta finansējums Līguma izpildei).

7.10.6. pēc Līguma noslēgšanas atklājas, ka, iesniedzot piedāvājumu iepirkumam, Izpildītājs ir apzināti sniedzis nepatiesu informāciju;

7.11. Ja Izpildītājs laužLīgumu pēc savas iniciatīvas un Pasūtītājs ir pienācīgi pildījis visas savas līgumsaistības, tad Izpildītājs maksā Pasūtītājam līgumsodu, kas sastāda 10 % (desmit procenti) no Līgumcenas.

7.12. Līgumsoda samaksa neatbrīvo Puses no Līguma saistību izpildes un Puses var prasīt kā līgumsoda, tā arī Līguma noteikumu izpildīšanu.

7.13. Katras Puses atbildība Līguma ietvaros aprobežojas ar Līguma 2.1. apakšpunktā norādīto kopējo Līguma summu, izņemot gadījumus, kad vainīgā Puse rīkojusies ļaunprātīgi vai ar rupju nolaidību.

## 8. KONFIDENCIALITĀTES NOSACĪJUMI

8.1. Puses apņemas bez otras Puses iepriekšējas rakstveida piekrišanas neizpaust jebkādu informāciju par otru Pusi vai tā uzņēmumu, ko tās ieguvušas Līguma izpildes gaitā, tostarp, jebkādu informāciju, ko kāda no Pusēm sniedz viena otrai Līguma izpildes laikā vai arī tā atklājas, pildot darba pienākumus, kā arī jebkuru šīs informācijas daļu, t.sk., bet ne tikai informāciju par personas datiem, personas sensitīvajiem datiem, par otras Puses darbību, finanšu stāvokli, tehnoloģijām, t.sk. rakstisku, mutisku, datu formā uzglabātu, audio – vizuālu un jebkurā citā veidā uzglabātu informāciju, kā arī informāciju par Līgumu, tā noslēgšanu un izpildi, izņemot Līguma 8.5.apakšpunktā noteiktajos gadījumos. Šis nosacījums ir spēkā gan Līguma izpildes laikā, gan 3 (trīs) gadus pēc Līguma darbības termiņa izbeigšanās, t.sk. arī pēc pirmstermiņa Līguma attiecību izbeigšanas.

8.2. Fizisko personu datu apstrādi Izpildītājs apņemas veikt atbilstoši Līguma 4.pielikuma noteikumiem.

8.3. Izpildītājs un Pasūtītājs apņemas sniegt Līguma 8.1.apakšpunktā norādīto informāciju

sava uzņēmuma darbiniekiem tikai nepieciešamības gadījumā un tādā apjomā, kādā tas ir nepieciešams tikai Līguma izpildei.

8.4. Pušu pienākums ir nodrošināt, ka tās amatpersonas, darbinieki, konsultanti u.c. personas, kuras izmantos Pušu informāciju, tiks iepazīstinātas ar Līgumā paredzētajiem nosacījumiem par konfidencialitāti pirms attiecīgo darbu uzsākšanas un saņems un izmantos šo informāciju vienīgi Līguma izpildes nodrošināšanai un tikai nepieciešamajā apjomā, kā arī uzņemsies un ievēros vismaz tādas pašas konfidencialitātes saistības (rakstiski apliecinot konfidencialitātes prasību ievērošanu), kādas Pusēm ir noteiktas Līgumā.

8.5. Pušu informācijas izpaušana netiks uzskatīta par Līguma noteikumu pārkāpumu vienīgi šādos gadījumos:

8.5.1. informācija tiek izpausta pēc tam, kad tā kļuvusi publiski zināma vai pieejama neatkarīgi no Pusēm;

8.5.2. informācija tiek izpausta normatīvajos aktos noteiktajos gadījumos, apjomā un kārtībā.

8.6. Ja Līguma 8.1.apakšpunktā norādīto informāciju pieprasa institūcijas, kurām uz to ir likumīgas tiesības, jebkurai Pusei ir tiesības izpaust šādu informāciju bez otras Puses iepriekšējas atļaujas.

8.7. Ievērojot augstāk norādītos konfidencialitātes nosacījumus, Izpildītājam ir tiesības pēc saviem ieskatiem norādīt uz Izpildītāja pieredzi un prasmēm, informējot savus esošos un potenciālos klientus, ka Izpildītājs veic vai ir sniedzis pakalpojumus Pasūtītājam.

## 9. AUTORTIESĪBAS

9.1. Visi Sistēmā ievadītie un Sistēmas darba rezultātā iegūtie dati visos to formātos ir Pasūtītāja ekskluzīvs īpašums. Ar šo Pasūtītājs apliecina, ka tam pieder visas autora mantiskās tiesības uz Sistēmu, tādēļ izmaiņu veikšana nevar tikt uzskatīta par trešajām personām piederošo autortiesību aizskārumu.

9.2. Izpildītājs apliecina, ka ne Līguma izpildes process, ne arī tā laikā sagatavotie rezultāti vai to atsevišķas daļas, kas ir autortiesību objekti, nepārkāpj vai neaizskar nekādas trešo personu autortiesības, ko Līguma izpilde vai tā laikā sagatavotie rezultāti vai to atsevišķas daļas ietver vai varētu ietvert. Tādējādi Izpildītājs apliecina, ka tas ir ieguvis autortiesības, kas nepieciešamas Līguma izpildei un Līgumā noteikto darba rezultātu sagatavošanai, kā arī nodrošina Pasūtītājam pilnīgu rīcību ar Līguma izpildes laikā radītajiem rezultātiem to turpmākajā izmantošanā atbilstoši Līguma 9.5.apakšpunktā noteiktajam.

9.3. Gadījumā, ja pret Pasūtītāju tiks vērstas trešo personu prasības par autortiesību pārkāpumiem, kas tieši vai netieši saistītas ar Līguma izpildes laikā radītajiem darbu rezultātiem vai to atsevišķām daļām, Izpildītājs apņemas atbildēt par šīm trešo personu prasībām un atbrīvot Pasūtītāju no jebkādas atbildības šajā sakarā. Izpildītājam ir pienākums atlīdzināt Pasūtītājam visus un jebkādus izdevumus, kas tam radušies saistībā ar šādiem trešo personu prasījumiem.

9.4. Pusēm (t.sk. jebkurām trešajām personām) saglabājas autora mantiskās tiesības uz

jebkuriem dokumentiem, materiāliem, datiem vai programmatūru, kura ir tikusi izmantota Līguma izpildes ietvaros un uz kuru Pusēm vai trešajām personām ir bijušas autora mantiskās tiesības jau pirms Līguma spēkā stāšanās brīža.

9.5. Autora mantiskās tiesības uz Līguma izpildes ietvaros izstrādātajiem un Pasūtītājam piegādātajiem Sistēmas papildinājumiem pāriet Pasūtītājam ar brīdi, kad Sistēmas papildinājumi ir pilnībā piegādāti Pasūtītājam (ko apliecina abpusēji parakstītais nodošanas – pieņemšanas akts) un Pasūtītājs ir pilnībā norēķinājies ar Izpildītāju par tā veiktajiem darbiem Līguma izpildē.

## 10. GARANTIJAS SAISTĪBAS

10.1. Izpildītājam jānodrošina EURES moduļa garantija par saviem līdzekļiem 12 (divpadsmit) mēnešu laikā. Garantijas termiņš tiek skaitīts no 3.posma darbu nodošanas – pieņemšanas akta parakstīšanas dienas.

10.2. Garantija ietver EURES moduļa bezatteices darbību pilnībā un attiecas gan uz Izpildītāja izstrādāto (izmainīto) programmatūru, gan uz programmatūru, kuras darbību ietekmē Izpildītāja izstrādātā (izmainītā) programmatūra.

10.3. Garantija ietver EURES moduļa kļūdu bezmaksas novēršanu šādos termiņos:

10.3.1. 1.kategorijas kļūda (avārija) – 8 (astoņu ) stundu laikā, skaitot darba laikā, darba dienās.

10.3.2. 2. kategorijas kļūda (kļūda, kuru nevar apiet) – 16 (sešpadsmit ) stundu laikā, skaitot darba laikā, darba dienās.

10.3.3. 3. kategorijas kļūda (kļūda, kuru var apiet) – 40 (četrdesmit ) stundu laikā, skaitot darba laikā, darba dienās.

## 11. NEPĀRVARAMA VARA

11.1. Neviena no Pusēm nav atbildīga par Līguma saistību neizpildi, ja saistību izpilde nav bijusi iespējama nepārvaramas varas apstākļu dēļ, kas radušies pēc Līguma noslēgšanas, un ja Puse par šādu apstākļu iestāšanos ir informējusi otru Pusi 5 (piecu) darba dienu laikā no šādu apstākļu rašanās dienas. Šajā gadījumā Līgumā noteiktais izpildes un samaksas termiņš tiek pagarināts attiecīgi par tādu laika periodu, par kādu šie nepārvaramās varas apstākļi ir aizkavējuši Līguma izpildi, bet ne ilgāk par 10 (desmit) kalendārajām dienām.

11.2. Ar nepārvaramas varas apstākļiem jāsaprot dabas stihijas (plūdi, vētras postījumi), katastrofas, streiki, karadarbība vai manevri, pilnvaroto institūciju izdotie normatīvie akti, kā arī citi tamlīdzīgi apstākļi, kuri tiek klasificēti pēc vispārpieņemtiem standartiem kā nepārvaramas varas apstākļi un kuru dēļ Līguma nosacījumu izpilde nav iespējama, un kurus Pusēm nebija iespējas ne paredzēt, ne novērst. Pusei, kura atsaucas uz nepārvaramas varas apstākļiem, ir jāpierāda, ka tai nebija iespēju ne paredzēt, ne novērst radušos apstākļus, kuru sekas, par spīti īstenotajai pienācīgajai rūpībai, nav bijis iespējams novērst.

11.3. Gadījumā, ja nepārvaramas varas apstākļi turpinās ilgāk kā 10 (desmit) kalendārās dienas, katra no Pusēm ir tiesīga izbeigt Līgumu, par to rakstveidā brīdinot otru Pusi 3 (trīs) kalendārās dienas iepriekš.

11.4. Puse, kuras saistību izpilde kļūst neiespējama Līguma 12.2.apakšpunktā norādīto apstākļu iestāšanās dēļ, brīdina par to otras puses kontaktpersonu, kura norādīta Līguma 6.punktā personīgi, telefoniski un rakstiskā veidā, norādot šo apstākļu prognozējamo ilgumu, un vienojas par tālāko sadarbības veidu

## 12. CITI NOTEIKUMI

12.1. Līgums stājas spēkā ar tā abpusēju parakstīšanas dienu un ir spēkā līdz Pušu saistību pilnīgai izpildei.

12.2. Izmaiņas un papildinājumi Līgumā ir spēkā tikai tad, ja tie noformēti rakstveidā un apstiprināti ar abu Pušu parakstiem.

12.3. Visi pēc Līguma spēkā stāšanās sastādītie Līguma grozījumi vai papildinājumi, ja tie sastādīti, ievērojot Līguma 12.2.apakšpunkta noteikumus, ir Līguma neatņemama sastāvdaļa.

12.4. Jebkuru pretrunu gadījumā jebkuros datos, noteikumos vai nosacījumos jebkurā no Līguma nosacījumiem, no vienas puses, un uz Līguma pamata veikto Pasūtījumu datos, noteikumos vai nosacījumos, no otras puses, informācija, ko satur Līguma nosacījumi, ir noteicošā, ja vien Pasūtījumā tieši nav noteikts pretējais.

12.5. Visi strīdi un domstarpības, kas varētu rasties starp Pusēm Līguma izpildes rezultātā, tiek risināti sarunu ceļā. Ja savstarpēja vienošanās netiek panākta 30 (trīsdesmit) dienu laikā, Pusēm ir tiesības griezties tiesā normatīvajos aktos noteiktajā kārtībā.

12.6. Ja Līguma darbības laikā notiek Puses reorganizācija, tās tiesības un pienākumus realizē tiesību un saistību pārņēmējs.

12.7. Ja kāds Līguma nosacījums zaudē spēku atbilstoši normatīvajiem aktiem, Līgums savu spēku nezaudē un attiecīgais nosacījums ir aizstājams ar spēkā esošu nosacījumu.

12.8. Ja kāds Līguma nosacījums zaudē spēku atbilstoši normatīvajiem aktiem un tas būtiski ietekmē Līgumā noteikto, Puses nekavējoties veic pārrunas un pieņem lēmumu par Līguma turpināšanas nosacījumiem vai arī par Līguma izbeigšanu.

12.9. Līguma izpildes ietvaros visa oficiālā sarakste, paziņojumi un informācija nogādājama personīgi vai sūtāma otrai Pusei pa pastu uz juridisko adresi, elektronisko pastu vai arī elektroniski Līguma 5.2. un 5.3.apakšpunktos norādītajiem Pušu pārstāvjiem.

12.10. Jautājumi, kas Līguma ietvaros nav noteikti vai atrunāti, tiek risināti saskaņā ar spēkā esošajiem normatīvajiem aktiem.

12.11. Līgums sastādīts latviešu valodā 2 (divos) identiskos eksemplāros, katrs uz 10 (desmit) lapām, un tam līguma noslēgšanas brīdī ir 5 (pieci) pielikumi uz 42 (četrdesmit divas) lapām. 1 (viens) Līguma eksemplārs glabājas pie Pasūtītāja, bet otrs – pie Izpildītāja, un tiem abiem ir vienāds juridiskais spēks.

12.12. Līgumam tā noslēgšanas brīdī ir pievienoti šādi pielikumi:

*1.pielikums – Tehniskā specifikācija;*

*2.pielikums – Finanšu piedāvājums;*

*3.pielikums – Izpildītāja piedāvāto vadošo speciālistu saraksts;*

*4.pielikums – Vienošanās protokols par fizisko personu datu apstrādi.*

*5.pielikums - Anketa par fizisku personu datu apstrādātāja tehniskajiem un organizatoriskajiem pasākumiem datu aizsardzības nodrošināšanai*

## 13. PUŠU REKVIZĪTI

**PASŪTĪTĀJS:**

**Nodarbinātības valsts aģentūra,**
Reģ.nr: 90001634668
K. Valdemāra ielā 38 k-1, Rīgā, LV – 1010
Banka: Valsts kase
SWIFT: TRELLV22
Konta Nr. LV06TREL2180451041000

/Kristīne Stašāne/

**IZPILDĪTĀJS:**

SIA Uniso
Reģ.nr. 40003562863
Valdlauči 1 - 40, Ķekavas pag., Ķekavas nov.
Konts: LV21HABA0551001217765

/Mārtiņš Rumkovskis/

**1. pielikums**
**pie 2018.gada 15.augusta līguma**
***Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrādi***

**Tehniskā specifikācija**

DG EMPL

# New EURES Regulation

## NCO Default Implementation Modules V0.81

arms Developments

# Table of Contents

# Abbreviations and Acronyms

| Abbreviation | Meaning |
|---|---|
| API | Application Programming Interface |
| CV | Curriculum Vitae |
| DB | Database |
| DDL | Data Definition Language |
| HTTPS | Hypertext Transfer Protocol Secure |
| JAR | Java Archive |
| JNDI | Java Naming and Directory Interface |
| JV | Job Vacancy |
| NCO | National Coordination Office |
| PES | Public Employment Service |
| PrES | Private Employment Service |
| REST | Representational State Transfer |
| SQL | Structured Query Language |
| XML | Extensible Markup Language |
| WAR | Web application Archive |

*Table 1 - Abbreviations and acronyms*

# Reference Documents

This section contains the list of all referenced documents. When referring to any of them, the bracketed reference will be used in the text, such as [R01].

| Ref. | Title | Reference | Version | Date |
|---|---|---|---|---|
| R01 | EURES New regulation | http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L:2016:107:TOC | N/A | N/A |
| R02 | EURES formats and standards specification - part 1<br>Job vacancy data standard description | EFSS_JV | 1.03 | 24/11/2017 |
| R03 | EURES formats and standards specification - part 2<br>Job seeker profile data standard description | EFSS_JSP | 1.03 | 24/11/2017 |
| R04 | EURES Functional Message Exchange Specifications | EURES-2018-INPUT-API-FMES | 1.3.1 | 11/07/2017 |
| R05 | Query mapping document | N/A | 5.10 | N/A |

*Table 2: Reference Documents*

# 1 PURPOSE OF THE DOCUMENT

This document serves these purposes:

- Provide a high-level description of which IT obligations of the Member States could be covered by the "NCO default implementation modules" offered by DG EMPL;
- Describe the intermediate repository, which is a database required for the use of each module;
- Describe the different modules and some possible implementation possibilities;
- List the technical requirements for the intermediate repository and for each module;
- Provide installation and configuration instructions;
- Provide instructions on the operational usage of the intermediate repository and of each module in case it provides a user interface, admin features or logs to consult;
- Provide an overview of the compatible and tested environments.

The first part of this document is targeted at IT decision makers to determine whether the provided modules should/could be used for their organisation.

The requirements allow determining the feasibility of each module within your infrastructure.

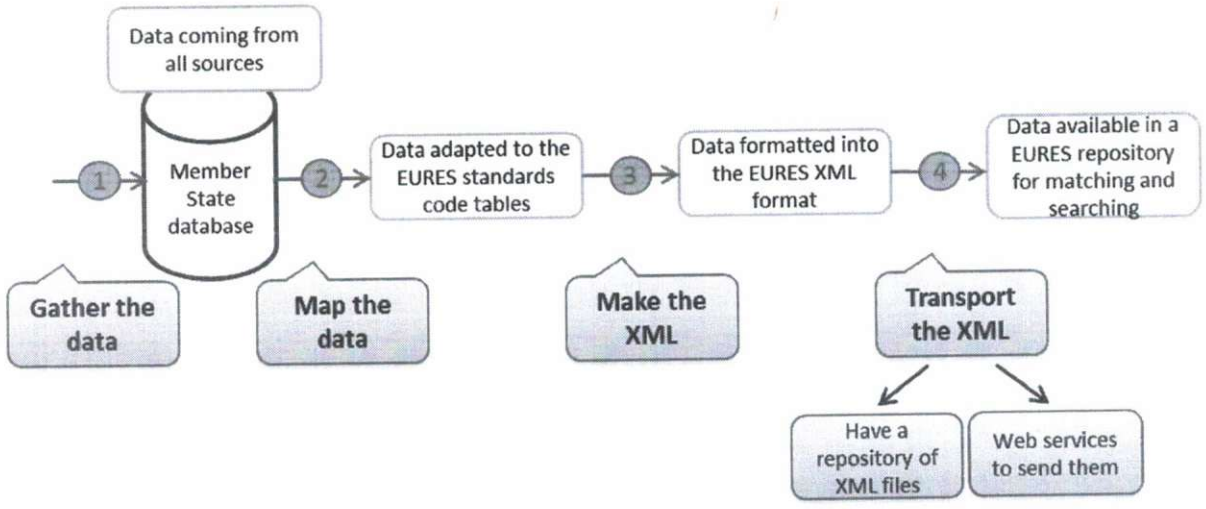
The last parts are targeted at IT operators that need to install and operate the modules.

# 2 OBLIGATIONS COVERED BY THE MODULES

Member states have the obligation to transfer job vacancies and jobseeker profiles (CVs) to DG EMPL [R01]. They must implement the "Uniform exchange system" which is described by several specification documents [R02][R03][R04]. The modules offered by DG EMPL provide support to implement parts of those documents.

The Member States have an obligation to produce a valid XML file of each job vacancy and jobseeker profile according to the EURES Standards and Formats specifications [R02][R03] and to transport it. This can be detailed in these steps:

1. You need to aggregate the records and the metadata of your sources (i.e. of your PESs and PrESs) into the Member State database;
2. You need to map the data from the codifications used in the Member State database to the code tables that are required by the EURES standards;
3. You need to build valid XML files that comply with the structure required by the EURES standards, including cardinality and business rules;
4. You need to maintain an up to date inventory of these XML files to be transferred to DG EMPL on request. This includes keeping timestamps of creation and modification of these XML files, as well as a short-term inventory of the unique ID's of closed job vacancies and/or jobseeker profiles;
5. Finally, you need to implement the web services described in the new EURES Functional Message exchange specifications [R04] that will transfer the inventory of (changed) records ID's and/or their XML files to DG EMPL.

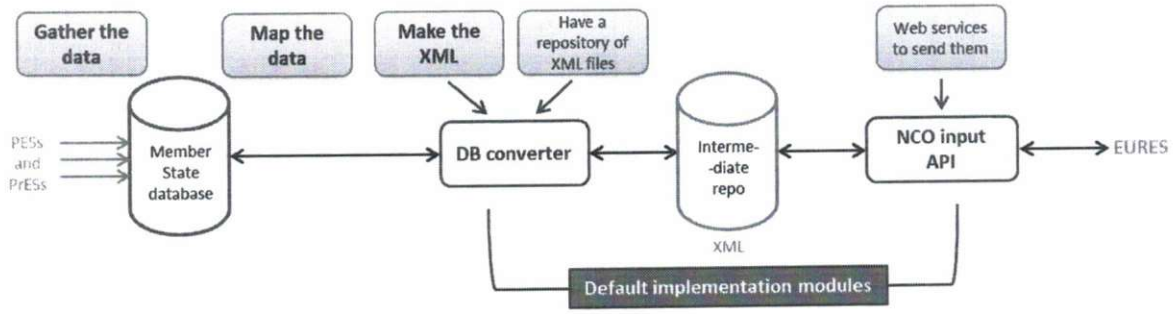



The modules offered by DG EMPL can cover a large part of these obligations if they are installed and configured in a proper way.

The **gathering** of the data of the sources and the **mapping** of this data are **not** covered by the modules. This will remain an obligation for which each Member State must implement its own solution.

The modules however are able to form a valid XML file with the mapped data, and maintain the repository of these files in an inventory, called the intermediate repository, with the required metadata. Finally, they

implement all the needed web services to transfer the ID's and XML files from the intermediate repository to DG EMPL.




**Important note: the modules are offered by DG EMPL to support the Member States. The usage of these modules does not remove the obligations on the Member States. In case one of the modules cannot be deployed and/or does not function as expected in the specific IT environment of your organisation, you will have to develop your own solution for the module that fails.**

The default implementation is designed for maximum modularity to allow a Member State to benefit the most of each solution offered, but also to provide the mitigation path in case one module fails.

**This version of default implementation is dealing with the transfer of JVs and Jobseeker Profiles (CVs).**

# 3 DEFAULT IMPLEMENTATION DESCRIPTION

## 3.1 INTERMEDIATE REPOSITORY

The NCO default implementation requires the configuration of an **intermediate repository** between the Member State database and the EURES database. This repository is deployed in the environment of the Member State. Its purpose is to store the records of the Member State in the XML format together with the metadata.

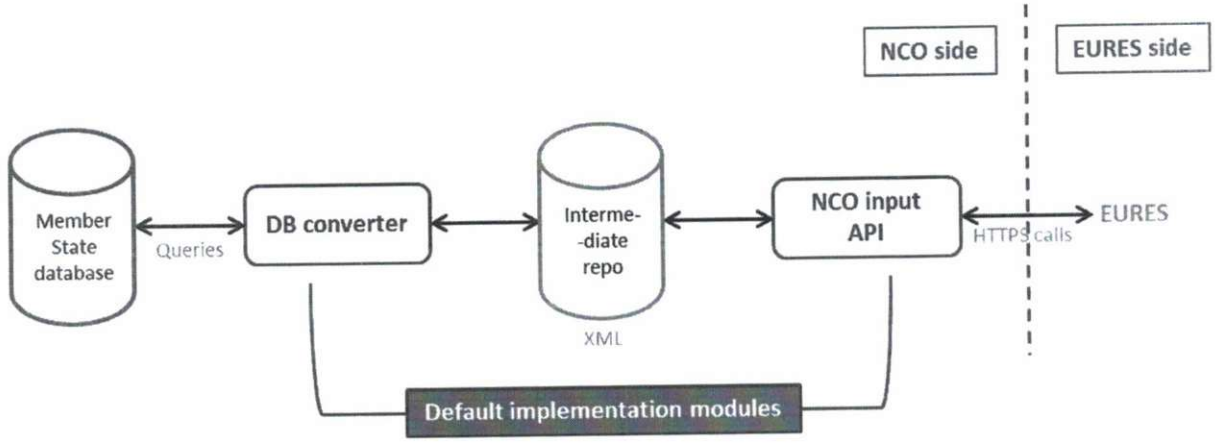
Note that the requirements for the structure of the intermediate repository can evolve over time. A version will therefore be associated to this repository, or, more precisely, to its structure. The use of the latest version can be mandatory or not, depending on the changes made. Also, the version of the intermediate repository will probably affect the versions needed for the default implementation modules.

For each intermediate repository version, the scripts enabling to configure this database will be provided to the Member States.

## 3.2 MODULES

The default implementation is made of two modules: the **DB converter module** and the **NCO input API module**.

The role of these modules is depicted on the figure below and their functionalities are described in the next sections.




Note that the modules of the default implementation will evolve over time and new functionalities will be added. A version will thus be associated to each module, the use of the latest one being mandatory or not, depending on the modifications made. New modules could also be implemented in the future.

### 3.2.1 DB Converter Module

The DB converter module allows extracting the records data from the Member State database, converting it into the EURES XML format and storing the result into the intermediate repository.

To use it, the Member State will therefore need to configure some queries to extract the data from its database [R05]. In these queries, the codes of the codifications defined in the EURES standards must be provided and not the ones of the national codifications of the Member State. An easy way to achieve this objective is to create mapping tables between the national codifications and the EURES standards ones. In so doing, these mapping tables can be directly used in the queries to return the desired code in the EURES standards codification.

Note that, in order to improve the performances of the DB converter module, the conversion of data into the XML format is multi-threaded.

This module also implements the synchronization of data between the Member State database and the intermediate repository. It will in fact regularly extract the records that have been created, modified or closed since the last synchronization, and update the intermediate repository to reflect the data of the Member State database. For this purpose, the Member State will have to provide in their database the date and time corresponding to the creation, last modification and closing of each record inside this database. Note that the period of synchronization between the Member State database and the intermediate repository is configurable.

The DB converter module also handles the metadata associated to the records. First of all, it replicates the one contained in the Member State database (as the record reference, the timestamps or the status for example), which should have the imposed format. Moreover, to implement correctly the NCO input API, the intermediate repository must contain the creation timestamp corresponding to the creation of the record in this repository. It must also contain the last modification and closing timestamps corresponding to the last modification and to a potential closing in this repository. The DB converter module will therefore add this information in the intermediate repository.

The first versions of this module will only produce XML files containing EURES technical minimum fields and EURES conformant mandatory fields for job vacancies, in one language. The management of the optional fields and the possibility to give the translated content of some fields will be available later on.

Furthermore, in the future versions of the DB converter module, the user will have to possibility to enable an option allowing triggering a call to the EURES input API to request a replication each time a change has been made in the intermediate repository. Doing so, he can avoid waiting for the next call to the NCO input API in order to synchronize its data. This option will be configurable via a property.

Finally, the errors related to the XML generated in the intermediate repository and to the corresponding metadata will be reported in the log files of future versions of the module.

### 3.2.2 NCO Input API Module

The purpose of the NCO input API module is to implement the API services that must be provided by the Member States and that will be called by the EURES system to replicate its job vacancies and jobseeker profiles. This module will query the intermediate repository to retrieve the needed data. Note that no conversion will be made since this repository will already contain the data in the format required for the NCO input API.

# 4 IMPLEMENTATION POSSIBILITIES

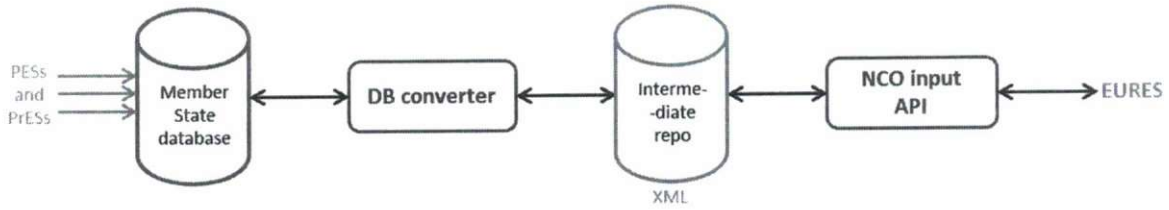
The modules can be used and combined in several ways. This chapter describes some possibilities of implementation of the Member State solution that make use of the NCO default implementation. Those possibilities can be combined.

For each solution, the versions of the modules used do not need to be the same. They must however support the same version of the intermediate repository structure.

In the case where the Member State decides to not use a module and to implement its own solution for this module instead, it is its own responsibility to implement a solution in line with the version used for the intermediate repository. Adjustments to the Member State's solution can be necessary if the structure of the intermediate repository changes.

## 4.1 DEPLOYING BOTH MODULES

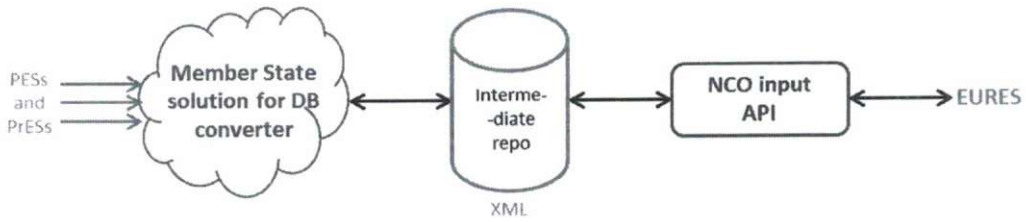
The default solution consists in using the DB converter module together with the NCO input API module. In this case, the Member State gathers the records of its sources in its database and deploys the intermediate repository. The modules are then used to connect the two databases together and to transmit records to DG EMPL.




## 4.2 DEPLOYING THE NCO INPUT API MODULE ONLY

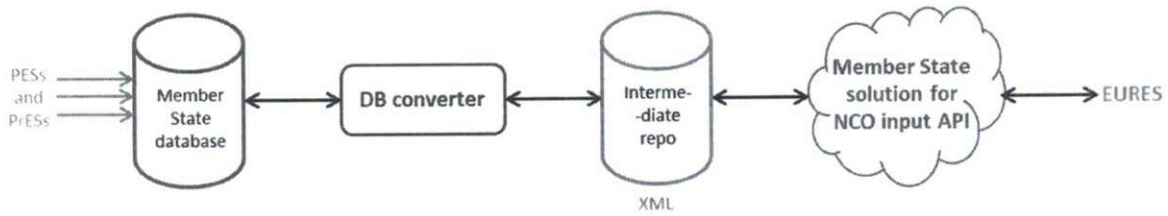
The Member State can choose to use only the NCO input API module. In this case, it deploys the intermediate repository and stores the XML records and the corresponding metadata in this repository using its own solution. The NCO input API is then deployed to implement the services needed for the transmission of data.

When developing its solution for the DB converter module, the Member State must make sure that all the metadata needed in the responses to the calls to the NCO input API is contained in the intermediate repository in the appropriate format.



## 4.3 Deploying the DB Converter Module Only

The Member State can choose to use only the DB converter module. In this solution, it deploys the DB converter module and the intermediate repository, and implements the NCO input API services that will allow retrieving the data from this database.

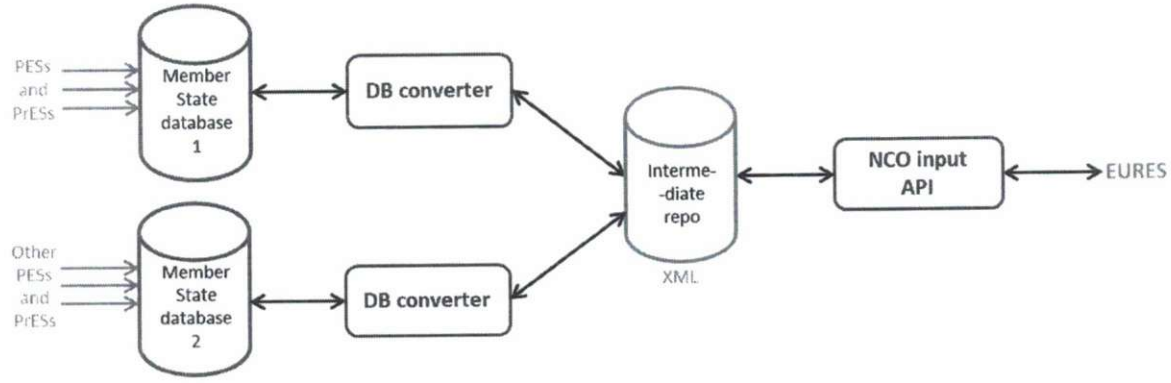



## 4.4 Deploying Multiple DB Converter Modules

The Member State has the possibility to have multiple databases and deploy one DB converter module for each of them to fill the intermediate repository.

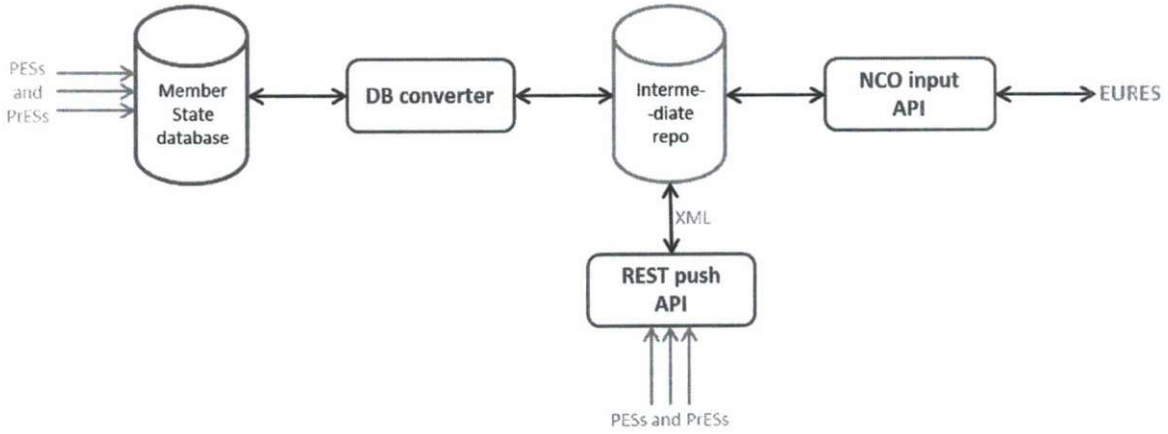
In this case, those databases have to contain distinct records and the records references have to be unique among the references of all the Member State databases.

Note that applying this strategy should not significantly improve the performances of the XML conversion since the DB converter module is already multi-threaded.



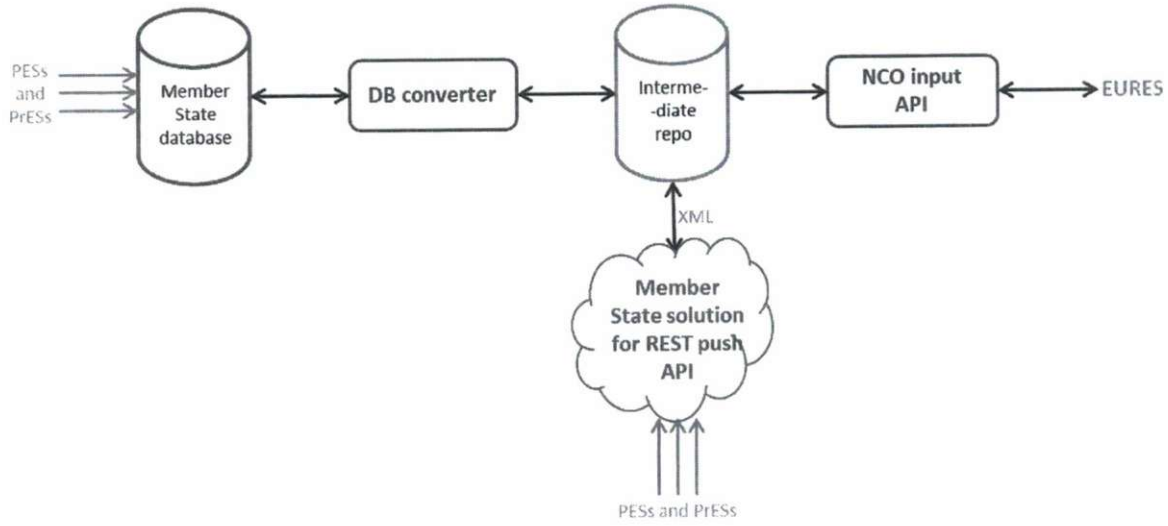

## 4.5 Deploying the REST Push API Module

Depending on the needs and requirements of the Member States, a REST Push API module **could** be created in the future to permit to directly push into the intermediate repository the records of one or more sources in the EURES XML format. This module could be used without the other ones.



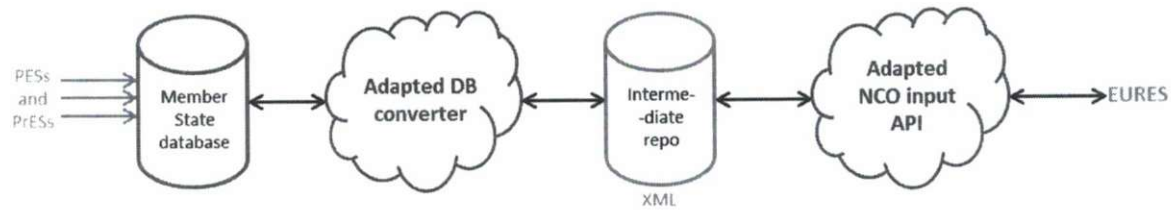

## 4.6 Allowing Direct Writing in the Intermediate Repository

The Member State could allow some sources to insert their data directly into the intermediate repository by developing its own solution for the REST push API module.




## 4.7 Adapting the Code

The sharing of the source code of the modules with the Member States is currently being investigated and is therefore **not** guaranteed. It would permit to adapt the modules of the default implementation to be more adjusted to the strategy and the system of the Member State.



# 5 PREREQUISITES

## 5.1 INTERMEDIATE REPOSITORY

The intermediate repository can either be a MongoDB or an SQL database (Oracle, PostgreSQL, or MySQL[1]). For the MongoDB, no setup is required. However, for using an SQL database, some provided scripts must be executed before the default implementation modules can be used.

## 5.2 DB CONVERTER MODULE

The use of the DB converter module is only possible if the following conditions are met:

- The Member State database must have an environment supporting Java 8. Indeed, the source code of the NCO default implementation is written using version 8 of Java. Some features implemented in Java 8 are used in this code. The NCO default implementation is therefore not backward compatible with previous Java versions;
- The Member State database must be an SQL database. It must of course contain the records mandatory fields and the needed metadata (such as the creation, last modification and closing timestamps for example);
- As mentioned in the previous section, the intermediate repository must be deployed as an SQL or a MongoDB database. It must be properly configured to contain the records mandatory fields and the needed metadata.

## 5.3 NCO INPUT API MODULE

The use of the NCO input API module is only possible if the following conditions are met:

- The Member State database must have an environment supporting Java 8. Indeed, the source code of the NCO default implementation is written using version 8 of Java. Some features implemented in Java 8 are used in this code and the NCO default implementation is therefore not backward compatible with previous Java versions;
- The intermediate repository must be deployed as an SQL or a MongoDB database. It must of course be configured to contain the records mandatory fields and the needed metadata.

[1] MySQL is not currently supported as an intermediary repository but it could be in a future version of the default implementation if requested (see chapter 8 for more details)

# 6 INSTALLATION AND CONFIGURATION

## 6.1 INTERMEDIATE REPOSITORY

The installation of the intermediate repository is mandatory to be able to use any of the modules provided by the NCO default implementation. The installation depends on the database technology chosen by the Member State to implement the intermediate repository. The Member State can choose between a MongoDB (noSQL) and one of the two following SQL database systems: Oracle, PostgreSQL or MySQL.

**Using MongoDB**

Installation instructions for MongoDB can be found at the official MongoDB installation manual. No scripts must be executed as the collections in the MongoDB will be created automatically by the modules on start up.

With the release 0.6.0, the collections jvDetail and jvMetadata can be deleted from the intermediate repository since these collections are now named recordDetail and recordMetadata. A reset should be performed to be resynchronized with the source.

**Using an SQL database**

To use an SQL database, the following scripts of the specific database system, located in eures-reg2018-nco-int-repo-0.6.2-sql, must be executed in the given order in the database:

1. <dbsystem>_IREP_TABLES_changeset-0.0.0.sql
2. <dbsystem>_IREP_TABLES_changeset-0.1.0.sql
3. <dbsystem>_IREP_TABLES_changeset-0.4.0.sql
4. <dbsystem>_IREP_TABLES_changeset-0.6.0.sql

where <dbsystem> is ORACLE, POSTGRESQL, or MYSQL. Make sure that the used database system is case insensitive (by default or by configuring it).

The execution of the 0.6.0 changeset removes the contents of the tables of the intermediate repository. A reset should be performed to be resynchronized with the source.

## 6.2 DB CONVERTER MODULE

In order for the DB converter module to work properly, a configuration file "*application.properties*" must be created first (included in the deliverable). The location of the file depends on the deployment strategy, covered in section 6.2.5.

All the configuration properties described in the next subsections must be added/updated in this "*application.properties*" file.

### 6.2.1 Global configuration

*Table **3*** summarizes the properties to globally configure the DB converter module. Optional properties (i.e. a default value exists) are suffixed with *(optional)*.

| Property | Description | Example |
|---|---|---|
| **logging.file** | The location of the file used for logging. If the file doesn't exist, it will be created. | /logs/db-converter.log |
| **jv.sync.batch.frequency.cron (optional)** | Cron expression altering the frequency with which the JVs in the intermediate repository are synced with the source database. The expression denotes the times at which a new sync is started. If the property is not specified, the batch for the JVs will not run. The example specifies a sync is started every thirty minutes on the first second of that minute.<br>To disable the synchronisation of JVs, this property must be removed from the properties file (or commented out by using the "#" prefix). | 0 0/30 * * * * |
| **cv.sync.batch.frequency.cron (optional)** | Cron expression altering the frequency with which the CVs in the intermediate repository are synced with the source database. The expression denotes the times at which a new sync is started. If the property is not specified, the batch for the CVs will not run. The example specifies a sync is started every thirty minutes on the first second of that minute, starting at the fifth minute of the hour.<br>To disable the synchronisation of CVs, this property must be removed from the properties file (or commented out by using the "#" prefix). | 0 5/30 * * * * |
| **sync.thread.pool.size** | The number of threads in the thread pool used for the synchronization operations between the Member State database and the intermediate repository.<br>Default if not provided: 10 | 10 |
| **nco.source.db.id** | In case of multiple DB converters deployed, it represents the ID of the DB converter. Not mandatory in case of :<br>- a single DB converter for both JVs and CVs;<br>- a single DB converter for the JVs and a single one for the CVs;<br>Mandatory in case of several DB converters containing the same type of records (JVs or CVs). | db-conv-1 |
| **management.port (optional) (needed if endpoints.enabled is set to true)** | The management port used for reaching the monitoring endpoints. | 8082 |
| **endpoints.enabled (optional)** | Property that allow enabling or disabling the monitoring endpoints. | false |

| | | |
|---|---|---|
| **endpoints.*.enabled (where * is an endpoint)[2] (optional)** | Property that allow enabling or disabling a specific monitoring endpoint (/health for example). | true |

*Table 3: Global Configuration of the DB converter module*

### 6.2.2 Configuration of the Source Database

The database connection can be configured either via a JNDI name or using the direct configuration. *Table 4* and *Table 5* describe the properties required respectively for a configuration using JNDI datasource and using a direct datasource.

Only one configuration is required. If both configurations are set, the JNDI datasource will be used.

| Property | Description | Example |
|---|---|---|
| **nco.source.db.jndi-name** | The JNDI name for the source database (if used). | java:comp/env/jdbc/ncodb |

*Table 4: Configuration of the database connection using a JNDI datasource*

| Property | Description | Example |
|---|---|---|
| **nco.source.db.url** | The Member State database URL (if not configured through JNDI). | jdbc:oracle:thin:@member-state.lan:1341:ORCL |
| **nco.source.db.username** | The Member State database username (if not configured through JNDI). | user |
| **nco.source.db.password** | The Member State database password (if not configured through JNDI). | password |
| **nco.source.db.driverClassName** | The database driver for the Member State database. | oracle.jdbc.OracleDriver |

*Table 5: Configuration of the source database connections using a direct datasource*

### 6.2.3 Configuration of the Intermediate Repository

The following property must be set for the module to be able to determine which type of database system will be used as intermediate repository (the one installed on the environment of the Member State in the previous section):

| Property | Description | Example |
|---|---|---|
| **spring.profiles.active** | This property is used by the module to determine the technology used for accessing the intermediate repository. If an SQL database is used as intermediate repository, this property must be set to "jpa". If using MongoDB, this property must be set to "mongo". | jpa |

*Table 6: Configuration of the type of database used for the intermediate repository*

[2] More information here: https://docs.spring.io/spring-boot/docs/current/reference/html/production-ready-endpoints.html

Depending on the chosen database system for the intermediate repository, some specific properties must be set:

**When using MongoDB**

Only use this configuration when the property the following property is set: "*spring.profiles.active=mongo*".

| Property | Description | Example |
|---|---|---|
| **spring.data.mongodb.host** | The hostname of the MongoDB the module will connect to. | localhost |
| **spring.data.mongodb.port** | The server port for the MongoDB. | 27017 |
| **spring.data.mongodb.database** | The name of the database that will be used in the MongoDB. | jvdb |
| **spring.data.mongodb.username** | The login user of the MongoDB server. | user |
| **spring.data.mongodb.password** | The login password required to login to the MongoDB. | password |

*Table 7: Configuration of the intermediate repository using mongo*

**When using an SQL database**

Only use this configuration when the following property is set: "*spring.profiles.active=jpa*".

The connection to the intermediate repository can be configured either using a JNDI datasource or a direct datasource. *Table **8*** and *Table **9*** describe respectively how to configure the properties for a JNDI datasource and a direct datasource.

Only one configuration is required. If both configurations are set, the JNDI datasource will be used.

| Property | Description | Example |
|---|---|---|
| **nco.irep.db.jndi-name** | The JNDI name for the intermediate repository. | java:comp/env/jdbc/intrepo |

*Table 8: Configuration of the intermediate repository using JPA with a JNDI datasource*

| Property | Description | Example |
|---|---|---|
| **nco.irep.db.url** | The intermediate repository database URL. | jdbc:postgresql://member-state-host:5432/jvdb |
| **nco.irep.db.username** | The intermediate repository database username. | jvdb |
| **nco.irep.db.password** | The intermediate repository database password. | password |
| **nco.irep.db.driverClassName** | The database driver for the intermediate repository. | org.postgresql.Driver |

*Table 9: Configuration of the Intermediate repository using JPA with a direct datasource*

### 6.2.4 Configuration of the queries

Finally, to be able to get all necessary data from the Member State database, the module uses some queries that must be configured correctly in the "*application.properties*" file. These queries are described in the query mapping document [R05].

### 6.2.5 Deployment of the module

After the "*application.properties*" file is filled in correctly, the module can be started. This can be done in several ways:

#### *Executing the executable JAR*

On Unix/Linux systems a specially crafted WAR file can be directly executed: eures-reg2018-nco-db-converter-executable .jar. This will start an embedded application container using Java. Therefore, only Java is required.

The "*application.properties*" file must be located in the same folder as the executable war file.

#### Installing the module as a service on Unix

To install the module as a service on Unix, refer to the following guide: http://docs.spring.io/spring-boot/docs/current/reference/html/deployment-install.html.

The "*application.properties*" file must be located in the same folder as the executable war file.

#### Using the java command

On non-Unix systems the jar can be started using the command java –jar eures-reg2018-nco-db-converter-executable.jar.

The "*application.properties*" file must be located in the same folder as the executable jar file.[3]

#### Supplying database drivers

To be able to upgrade the JDBC drivers for the intermediate repository and not to limit the supported[4] database systems, the database drivers are not included in the war but need to be provided – as is standard practice when using an application server. To provide the database driver, place the necessary jar files in a folder. Provide this folder to the executable war by setting the environment variable "*LOADER_PATH*".

[3] When running the java –jar <jar> command, it MUST be run from the directory the jar and application properties reside in. Otherwise the application.properties file will not be detected correctly.

[4] Given the multitude of database systems available only the compatibility with a few major database drivers will be extensively tested.

## Example service setup on modern Linux

This example contains instructions on how to create a unit file for a systemd based Linux distribution[5]. This will create a service on the operating system for the database converter.

- Create the folder */opt/eures/db-converter*
- Copy the file eures-reg2018-nco-db-converter-executable.war to that folder.
- Copy the application.properties to that folder.
- Create the folder */opt/eures/db-drivers* and put the chosen database driver in that folder.
- In */etc/system/system* create a file named eures-db-converter.service with the content below.

```
[Unit]
Description=Eures database converter
After=syslog.target

[Service]
User=eures
Environment="LOADER_PATH=/opt/eures/db-drivers"
ExecStart=/opt/eures/db-converter/eures-reg2018-nco-db-converter-
executable.jar
SuccessExitStatus=143

[Install]
```

- Start the service by using *systemctl start eures-db-converter.service*
- To run the service at system startup execute the command *systemctl enable eures-db-converter.service*

## Running from the command line example

For testing it may be simpler to run the database converter in userspace from a normal, non-root account, before moving to a more robust setup as described in the previous section. Instructions to that end can be found below.

- Create a folder named *eures* in the home directory of the user.
- In that folder create subfolders named *db-drivers* and *db-converter*.
- In *db-drivers* put the jars for your database system.
- In *db-converter* put eures-reg2018-nco-db-converter-executable.war and the application.properties file
- On the command line cd into *db-converter*.
- Execute the command *export LOADER_PATH=$HOME/eures/db-drivers*
- Execute the *eures-reg2018-nco-db-converter-executable.jar* file.

### *Using Tomcat*

The artefact eures-reg2018-nco-db-converter.war can be deployed on a Java Servlet Specification v3 compliant application server like Apache Tomcat.

[5] Ubuntu, Red Hat Enterprise, Debian, CentOS, and SUSE Linux Enterprise server have adopted systemd as a default since v15.04,v7.0,v8,v7.14.04, and v12 respectively.

The Tomcat Webapp will need to know the location of the *"application.properties"* configuration file. The simplest way to achieve this is to configure it as an environment property.

In the Tomcat configuration directory, create a file with the same name as the war to be deployed in the directory conf/Catalina/localhost. For instance, when deploying *eures-reg2018-nco-db-converter.war*, create *eures-reg2018-nco-db-converter.xml*. The file needs to have the following content:

```
<Context><Environment name="spring.config.location" type="java.lang.String"
value="<path-to>/application.properties"/></Context>
```

The context element can also be added directly to the Host element of the conf/server.xml file with the following modification:

```
<Context path="eures-reg2018-nco-db-converter"><Environment
name="spring.config.location" type="java.lang.String" value="<path-
to>/application.properties"/></Context>
```

***Using Weblogic***

In Weblogic, an alternative is required as there is no simple way to specify environment properties. Therefore, start the Weblogic server with the following argument

- `Dspring.config.location=<some-path>`

where *"<some-path>"* must be replaced by the absolute path to the folder containing the *"application.properties"* file. Then deploy the application on the Weblogic application server.

# 6.3 NCO INPUT API MODULE

In order for the NCO input API module to work properly, a configuration file *"application.properties"* must be created first (included in the deliverable). The location of the file depends on the deployment strategy, covered in section 6.3.3.

All the configuration properties described in the next subsections must be added/updated in this *"application.properties"* file.

## 6.3.1 Global Configuration

*Table **10*** summarizes the properties to globally configure the Input API module.

| Property | Description | Example |
|---|---|---|
| **input.api.ping.message** | The message returned in the response to a call to the Ping service (see [R04]). | Hello from Input API |
| **logging.file** | The location of the file used for logging. If the file doesn't exist, it will be created. | /logs/input-api.log |
| **active.cv.controller (optional)** | Property that allows activating or deactivating the CV endpoint. If the property is not set, it will be deactivated. | true |
| **active.jv.controller** | Property that allows activating or deactivating | true |

| | | |
|---|---|---|
| **(optional)** | the JV endpoint. If the property is not set, it will be deactivated. | |
| **management.port (optional) (needed if endpoints.enabled is set to true)** | The management port used for reaching the monitoring endpoints. | 8081 |
| **endpoints.enabled (optional)** | Property that allow enabling or disabling the monitoring endpoints. | false |
| **endpoints.*.enabled (where * is an endpoint)[6] (optional)** | Property that allow enabling or disabling a specific monitoring endpoint (/health for example). | true |

*Table 10: Global Configuration of the Input API module*

### 6.3.2 Configuration of the Intermediate Repository

The same configuration as the one used for the intermediate repository in the DB converter module can be used. This configuration is entirely described in section 6.2.3.

### 6.3.3 Deployment of the module

After the "*application.properties*" file is filled in correctly, the module can be started. This can be done in several ways:

#### *Executing the executable JAR*

On Unix/Linux systems a specially crafted WAR file can be directly executed: *eures-reg2018-nco-input-api-executable.jar*. This will start an embedded application container using Java. Therefore, only Java is required.

The "*application.properties*" file must be located in the same folder as the executable war file.

#### Installing the module as a service on Unix

To install the module as a service on Unix, refer to the following guide: http://docs.spring.io/spring-boot/docs/current/reference/html/deployment-install.html.

The "*application.properties*" file must be located in the same folder as the executable war file.

#### Using the java command

On non-Unix systems the jar can be started using the command calling java –jar eures-reg2018-nco-input-api-executable.jar.

The "*application.properties*" file must be located in the same folder as the executable war file.[7]

#### Supplying database drivers

To be able to upgrade the JDBC drivers for the intermediate repository and not to limit the supported[8] database systems, the database drivers are not included in the war but need to be provided – as is standard

[6] More information here: https://docs.spring.io/spring-boot/docs/current/reference/html/production-ready-endpoints.html

[7] When running the java –jar <jar> command, it MUST be run from the directory the jar and application properties reside in. Otherwise the application.properties file will not be detected correctly.

practice when using an application server. To provide the database driver, place the necessary jar files in a folder. Provide this folder to the executable war by setting the environment variable "*LOADER_PATH*".

[8] Given the multitude of database systems available only the compatibility with a few major database drivers will be extensively tested.

## Example Setup on modern Linux

This example contains instructions on how to create a unit file for a systemd based Linux distribution[9].

- Create the folder */opt/eures/input-api*
- Copy the file eures-reg2018-nco-input-api-executable.jar to that folder.
- Copy the application.properties to that folder.
- Create the folder */opt/eures/db-drivers* and put the chosen database driver in that folder.
- In */etc/system/system* create a file named eures-input-api.service with the content below.

```
[Unit]
Description=Eures input api
After=syslog.target

[Service]
User=eures
Environment="LOADER_PATH=/opt/eures/db-drivers"
ExecStart=/opt/eures/input-api/eures-reg2018-nco-input-api-
executable.jar
SuccessExitStatus=143

[Install]
```

- Start the service by using *systemctl start eures-input-api.service*
- To run the service at system startup execute the command *systemctl enable eures-input-api.service*

## Running from the command line example

For testing it may be simpler to run the database converter in userspace from a normal, non-root account, before moving to a more robust setup as described in the previous section. Instructions to that end can be found below.

- Create a folder named *eures* in the home directory of the user.
- In that folder create subfolders named *db-drivers* and *input-api.*
- In *db-drivers* put the jars for your database system.
- In *db-converter* put *eures-reg2018-nco-input-api-executable.war* and the application.properties file
- On the command line cd into *db-converter*.
- Execute the command *export LOADER_PATH=$HOME/eures/db-drivers*
- Execute the *eures-reg2018-nco-input-api-executable.jar* file.

### *Using Tomcat*

The artefact eures-reg2018-nco--input-api.war can be deployed on a Java Servlet Specification v3 compliant application server like Apache Tomcat.

[9] Ubuntu, Red Hat Enterprise, Debian, CentOS, and SUSE Linux Enterprise server have adopted systemd as a default since v15.04,v7.0,v8,v7.14.04, and v12 respectively.

The Tomcat Webapp will need to know the location of the *"application.properties"* configuration file. The simplest way to achieve this is to configure it as an environment property.

In the Tomcat configuration directory, create a file with the same name as the WAR to be deployed in the directory conf/Catalina/localhost. For instance, when deploying *eures-reg2018-nco--input-api.war*, create *eures-reg2018-nco--input-api.xml*. The file needs to have the following content:

```
<Context><Environment name="spring.config.location" type="java.lang.String"
value="<path-to>/application.properties"/></Context>
```

The context element can also be added directly to the Host element of the *conf/server.xml* file with the following modification:

```
<Context path="eures-reg2018-nco-input-api"><Environment
name="spring.config.location" type="java.lang.String" value="<path-
to>/application.properties"/></Context>
```

In this case, the path provided can be an absolute path to the Tomcat directory or a relative path (that is relative to the Tomcat home directory).

### *Using Weblogic*

In Weblogic, an alternative is required as there is no simple way to specify environment properties. Therefore, start the Weblogic server with the following argument

```
-Dspring.config.location=<some-path>
```

where *"<some-path>"* must be replaced by the absolute path to the folder containing the *"application.properties"* file. Then deploy the application on the Weblogic application server.

# 7 OPERATING INSTRUCTION

## 7.1 INTERMEDIATE REPOSITORY

The contents of the intermediate repository can be checked using an interface tool to connect to the used database (e.g. SQL Developer, MongoDB Compass...).

### 7.1.1 Clearing the intermediate repository

**Warning:** The execution of the following commands will drop all the data inside intermediate repository. Use them carefully.

***MongoDB***

1. Make sure the DB converter module NCO Input API modules are not running;
2. Drop the database configured in the *spring.data.mongodb.database* using one of the following methods:
    a. Connect to the MongoDB database using a mongo client like *Robomongo*[10] and drop the database of the intermediate repository (see Figure 1 for an illustration in Robomongo);
    b. Connect to MongoDB database using mongo shell and execute the db.dropDatabase() command[11].
3. On the next sync, the database will be automatically recreated.

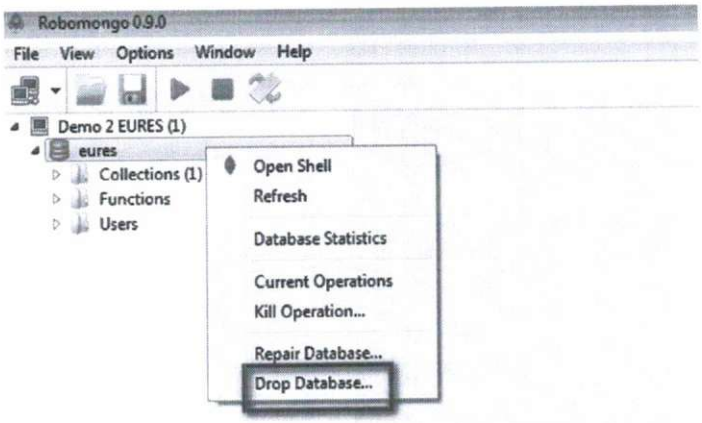



**Figure 1: Drop a database in Robomongo**

[10] https://robomongo.org
[11] https://docs.mongodb.com/manual/reference/method/db.dropDatabase/

***SQL Databases***

1. Make sure the DB converter module NCO Input API modules are not running;
2. Connect to the intermediate repo that is configured per the instructions of section 6.2.3;
3. Empty following tables in the order specified using TRUNCATE TABLE or DELETE FROM commands:
    - IREP_JV_DETAIL;
    - IREP_JV_METADATA;
    - IREP_SYNC_WARNINGS;
    - IREP_SYNC_EXCEPTION_DETAILS;
    - IREP_EXCEPTION_DETAILS;
    - IREP_SYNC.

## 7.2 DB CONVERTER MODULE

Administrative services for the DB converter module will be exposed on the following path.

```
{host}/db-converter/{type}/{service}
```

The type can be *jv* or *cv*. It indicates the type of records that will be synchronized.

The service can be *ping*, *reset* or *syncDbs*:

- The *ping* service can be used to check if the DB Converted module is deployed correctly;
- The *reset* service can be used to reset the intermediate repository, purging any JVs that have disappeared from the source database;
- The *syncDbs* service can be used to manually trigger synchronization between intermediate repository and source database.

Care must be taken not to expose these services to the outside world as this would allow unknown entities to interfere with the correct operation of the DB converter module. This can be easily checked using the ping service. The ping service can be accessed by entering {host}/db-converter/{type}/ping in any web browser. The request to ping should return an error if accessed from outside the premises of the NCO.

The log files for this module can either be found in the logging facility of the application server the module is deployed on or in the file specified in configuration (see section 6.2.1). In many application server implementations, the configuration of the logging facilities of the application server itself take precedence over the configuration specified in the configuration file of the application. This can influence the location of the log files.

## 7.3 NCO INPUT API MODULE

The services of the NCO input API will be available at the configured location at the following path:

```
{host}/input/api/{type}/v0.1/{service}
```

The type can be *jv* and *cv* and indicates the type of records for which information will be returned.

The *ping* service can be used to check if the NCO input API is correctly deployed, while the other services (*getAll*, *getChanges* and *getDetails*) can be used to check if it correctly transmits the data in the intermediate repository.

The log files for this module can either be found in the logging facility of the application server the module is deployed on or in the file specified in the configuration (see section 6.3.1). In many application server implementations, the configuration of the logging facilities of the application server itself take precedence over the configuration specified in the configuration file of the application. This can influence the location of the log files.

# 8 RELEASE NOTE

The following chapter presents a quick overview of the compatible and tested environments for the default implementation. The intermediate repository and the two modules are tackled separately.

The test and compatibility matrix for the 2 repositories (intermediate repository and source database) are displayed in respectively *Table **11*** and *Table **12***. The modules DB converter and Input API are displayed in respectively *Table **13*** and *Table 14*.

For each feature, the following values are displayed:

- Tested: displays whether the feature was fully tested, partially tested or not tested;
- Compatible: displays whether the feature is guaranteed to be compatible, should be compatible (i.e. not fully tested) or is not compatible (tests revealed bugs related to that feature).

| Feature | Tested | Compatible | Versions |
|---|---|---|---|
| **Use mongo DB** | Full | Yes | 3.2.x,3.4.x |
| **Use Oracle DB** | Full | Yes | 12c |
| **Use PostgreSQL** | Partial | Should | >9.3.x |
| **Use MySQL** | Partial | Yes | >5.7.x |
| **Use h2** | Full | Yes | |
| **Other SQL database**[12] | No | Should | |

*Table 11: Compatibility table for the intermediate repository*

| Feature | Tested | Compatible |
|---|---|---|
| **Use Oracle DB** | Full | Yes |
| **Use PostgreSQL** | No | Should |
| **Use MySQL** | No | Should |
| **Use h2** | Full | Yes |
| **Other SQL database** | No | Should |

*Table 12: Compatibility table for the source database*

| Feature | Tested | Compatible |
|---|---|---|
| **Execute executable WAR** | Full | Yes |
| **Install as a service in UNIX** | Partial | Yes |
| **Start using the java command** | Partial | Should |
| **Deploy in Tomcat** | Partial | Should |
| **Deploy in WebLogic** | Full | Yes |
| **Other application server**[13] | No | Should |

*Table 13: Compatibility table for the DB converter module*

[12] No scripts are provided to generate DDL for another SQL database. Therefore, if another database engine is to be used, the administrator might have to adapt the scripts generated for oracle DB, PostgreSQL, or MySQL for the target database engine.

[13] In order to deploy the WAR file in another application server, a deployment descriptor file might have to be provided.

| Feature | Tested | Compatible |
|---|---|---|
| **Execute executable WAR** | Full | Yes |
| **Install as a service in UNIX** | Partial | Yes |
| **Start using the java command** | Partial | Should |
| **Deploy in Tomcat** | Partial | Should |
| **Deploy in WebLogic** | Full | Yes |
| **Other application server**[14] | No | Should |

*Table 14: Compatibility table for the Input API module*

## 8.1 Functionalities Added in Default Implementation v0.3.0

The changes made to the default implementation modules between versions 0.2.0 and 0.3.0 can be summarized as follows:

- Adaptation of the URLs of the NCO input API according to version 1.20 of the FMES [R04];
- Adaptation of the JSON content returned by the services of the input API according to version 1.20 of the FMES [R04];
- Support of Gzip compression in the transferred data as described in the FMES [R04];
- Upgrade of Spring Boot version to 1.4.5.RELEASE.

Note that no change has been made to the DB converter module or to the intermediate repository scripts, so only the NCO input API module should be upgraded.

## 8.2 Functionalities Added in Default Implementation v0.4.0

The changes made to the default implementation modules between versions 0.3.0 and 0.4.0 can be summarized as follows:

- Updates of existing queries and adding of new queries to support the extraction of the fields in the Conformant Optional layer (see also [R03]) of the HR-XML standard (see also [R05]);
- Change of the way the status of a JV sync is saved in the intermediate repository from integer to String (text);
- Adding of SQL scripts for the intermediate repository to create indexes on some tables.

In the release 0.4.0, both DB converter and NCO input API modules are updated. Here is a summary of the operations to be performed to install the release 0.4.0 of the default implementation starting from release 0.3.0 if using a **MongoDB** as intermediate repository (to be executed in the provided order):

1. [Optional] Empty the intermediate repository (necessary because of the update of queries for Conformant Optional layer) as described in section 7.1.1. Alternatively, the *reset* service described in section 7.2 can be called after deployment to reset the database provided that the query *query.jv.getActiveIds* is properly configured (see step 3).;
2. Deploy the new DB converter and NCO input API modules;

[14] In order to deploy the WAR file in another application server, a deployment descriptor file might have to be provided.

3. [If step 1 not executed] Call the endpoint of the *reset* service described in section 7.2.

Here is a summary of the operations to be performed to install the release 0.4.0 of the default implementation starting from release 0.3.0 if using a **SQL database** as intermediate repository (to be executed in the provided order):

1. Empty the intermediate repository (necessary because of the update of queries for Conformant Optional layer) as described by section 7.1.1;
2. Execute the script <dbsystem>_IREP_TABLES_changeset-0.4.0.sql corresponding to the type of DB;
3. Deploy the new DB converter and NCO input API modules.

## 8.3 Functionalities Added in Default Implementation v0.5.0

The changes made to the default implementation modules between versions 0.4.0 and 0.5.0 can be summarized as follows:

- Database drivers are no longer supplied with the runnable jar. This allows more freedom to provide support for other database systems, as well as upgrading drivers without running into version conflicts with the supplied drivers. See sections 6.2.5 and 6.3.3 for details on how to provide the necessary drivers;
- Updates of existing queries and adding of new queries to support the extraction of all the fields for a JV (see also [R03]) of the HR-XML standard (see also [R05]);
- The intermediate repository now supports MySQL databases;
- Upgrade to the version 1.1 of the EURES standard.

In the release 0.5.0, both DB converter and NCO input API modules are updated. Here is a summary of the operations to be performed to install the release 0.5.0 of the default implementation starting from release 0.4.0 (to be executed in the provided order):

1. [Optional] Empty the intermediate repository as described in section 7.1.1. Alternatively, the *reset* service described in section 7.2 can be called after deployment to reset the database provided that the query *query.jv.getActiveIds* is properly configured (see step 3).;
2. Deploy the new DB converter and NCO input API modules;
3. [If step 1 not executed] Call the endpoint of the *reset* service described in section 7.2.

## 8.4 Functionalities Added in Default Implementation v0.6.0

The changes made to the default implementation modules between versions 0.5.0 and 0.6.0 can be summarized as follows:

- The possibility to have multiple source databases synchronized with the intermediate repository has been added (see section 4.4 for more details);
- The DB converter can fetch CVs containing a few fields, detailed in [R05]. It can be configured to fetch JVs and/or CVs, which, for example, would allow deploying two DB converters, one for the JVs and one the CVs;

- The synchronisation of JVs and CVs can de disabled by removing the relevant properties (see jv.sync.batch.frequency.cron and cv.sync.batch.frequency.cron properties in *Table **3*** for more details);
- The NCO input API can transmit information about CVs. As for the DB converter, it can activate or deactivate the JV and/or CV endpoints (see active.jv.controller and active.cv.controller properties in *Table **10*** for more details);
- The number of threads used when resetting or synchronizing the intermediate repository with the Member State database can be configured by means of a new property. If not configured, the value is set to 10 threads. Consequently, the Member States need to explicitly set it to 1 if they want to avoid multi-threading (see property sync.thread.pool.size in *Table **3*** for more details);
- The possibility to monitor the Input API and DB Converter modules using Spring Boot Actuator[15] (see new properties "management.port ","endpoints.enabled" and "endpoints.*.enabled" added in *Table **3*** and *Table **10***). Note that by default, this feature is disabled;
- Adding of SQL scripts for the intermediate repository to create indexes on some tables;
- Support for MongoDb 3.4.x;
- Upgrade to Spring boot 1.5.7.

Here is a summary of the operations to be performed to install the release 0.6.0 of the default implementation starting from release 0.5.0 (to be executed in the provided order):

1. if using an **SQL database** as intermediate repository, execute the script <dbsystem>_IREP_TABLES_changeset-0.6.0.sql corresponding to the type of DB (note that execution of this script will automatically remove all the JVs stored in the intermediate repository);
2. Add the new properties to the DB converter and to the NCO input API modules;
3. Deploy the new DB converter and NCO input API modules.

## 8.5 FUNCTIONALITIES ADDED IN DEFAULT IMPLEMENTATION V0.7.0

The changes made to the default implementation modules between versions 0.6.0 and 0.7.0 can be summarized as follows:

- Upgrade to the version 1.3 of the EURES standard (see [R02] and [R03]);
- Addition of privacy queries so that processing instructions can be added to the CV xml (see "Processing Instruction codes" sheet in [R05] for more information);
- Update of the queries (CV and JV) (see [R05] for more information. Two new Excel sheets have been added in the document listing the changes made in the queries).

In the release 0.7.0, the DB converter module is updated (Since the NCO input API module has not changed, the previous version can be used). Here is a summary of the operations to be performed to install the release 0.7.0 of the default implementation starting from release 0.6.0 (to be executed in the provided order):

1. [Optional] Empty the intermediate repository as described in section 7.1.1.

[15] More information about Spring Boot Actuator can be found on https://docs.spring.io/autorepo/docs/spring-boot/1.4.5.RELEASE/reference/html/production-ready-monitoring.html

2. Deploy the new DB converter;
3. [If step 1 not executed] Call the endpoint of the *reset* service described in section 7.2.

**2. pielikums**
**pie 2018.gada 15.augusta līguma**
***Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrādi***

**Finanšu piedāvājums**

| Izmaksas | Cena (EUR bez PVN) |
|---|---|
| Bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrāde, iepirkuma identifikācijas Nr. NVA 2018/19 | 41 900,00 |

**PASŪTĪTĀJS:**

**Nodarbinātības valsts aģentūra**,
Reģ.nr: 90001634668
K. Valdemāra ielā 38 k-1, Rīgā, LV – 1010

/Kristīne Stašāne/

**IZPILDĪTĀJS:**

SIA Uniso
Reģ.nr. 40003562863
Valdlauči 1 - 40, Ķekavas pag., Ķekavas nov.
Konts: LV21HABA0551001217765

/Mārtiņš Rumkovskis/

**3. pielikums**
**pie 2018.gada 15.augusta līguma**
***Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrādi***

## Izpildītāja vadošo darbinieku saraksts

"Bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrāde" iepirkuma identifikācijas Nr. NVA 2018/19

| **Loma projektā** | **Speciālista vārds, uzvārds** |
|---|---|
| Programmētājs | Guntis Ozols |
| Programmētājs | Jānis Birģelis |

**PASŪTĪTĀJS:**

**Nodarbinātības valsts aģentūra,**
Reģ.nr: 90001634668
K. Valdemāra ielā 38 k-1, Rīgā, LV – 1010

/Kristīne Stašāne/

**IZPILDĪTĀJS:**

SIA Uniso
Reģ.nr. 40003562863
Valdlauči 1 - 40, Ķekavas pag., Ķekavas nov.
Konts: LV21HABA0551001217765

/Mārtiņš Rumkovskis/

**4. pielikums**
**pie 2018.gada 15.augusta līguma**
***Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrādi***

**Līguma pielikums par fizisko personu datu apstrādi**

**Nodarbinātības valsts aģentūra**, nodokļu maksātāja reģistrācijas numurs: 90001634668, adrese: K. Valdemāra ielā 38 k-1, Rīgā, LV - 1010, tās direktores Evitas Simsones personā, kura rīkojas saskaņā ar Ministru kabineta 2012.gada 18.decembra noteikumiem Nr.876 „Nodarbinātības valsts aģentūras nolikums" (turpmāk – Pārzinis), no vienas puses, un

SIA Uniso, reģ.Nr. 40003562863, tās Valdes locekļa Mārtiņa Rumkovska personā, kurš darbojas uz statūtu pamata (turpmāk – Apstrādātājs), no otras puses,
turpmāk abi kopā saukti Puses, bet katrs atsevišķi – Puse), ņemot vērā to, ka lai Apstrādātājs pamatojoties uz Pušu starpā noslēgto līgumu Nr. _____ "Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas (BURVIS) uzturēšanu un papildinājumu izstrādi" (turpmāk – Līgums) Pārziņa interesēs un uzdevumā atbilstoši Līgumā noteiktajam nodrošinātu Līguma saistību izpildi, tajā skaitā izmaiņu pieprasījumu realizāciju un/vai problēmziņojumu risināšanu (novēršanu), Pārzinis atļauj (uztic) Apstrādātājam Eiropas Parlamenta un padomes 2016.gada 27.aprīļa regulas 2016/679 par fizisku personu aizsardzību attiecībā uz personas datu apstrādi un šādu datu brīvu apriti un ar ko atceļ Direktīvu 95/46/EK (Vispārīgā datu aizsardzības regula) (turpmāk – regula) izpratnē un personas datu operatoram – Fizisko personu datu aizsardzības likuma izpratnē, BURVIS sistēmā esošo personu datu apstrādi minimālajā apjomā, kas nepieciešams Līguma 1.1.apakšpunktā noteikto uzdevumu izpildei, tajā skaitā programmatūras izstrādei, testēšanai un/vai problēmziņojumu risināšanai (novēršanai), vienojas par turpmāko:

**1. Priekšmets**

1. Apstrādātājs veic Pārziņa interesēs un uzdevumā personas datu apstrādi, lai izpildītu saistības, kuras Apstrādātājs uzņēmies ar Pārziņa un Apstrādātāja starpā noslēgto Līgumu. Pārzinis pēc iespējas minimizē personas datu apjomu, ko Līguma izpildē apstrādā Apstrādātājs.
2. Apstrādātājs veic personas datu apstrādi, ievērojot personas datu aizsardzības principus un saskaņā ar spēkā esošajiem tiesību aktiem, tajā skaitā, regulu, Līgumu, šo vienošanos un Pārziņa tiesiskajām norādēm. Apstrādātājs neizmanto tam uzticētos personas datus no Līguma neizrietošiem nolūkiem (mērķiem), vai citādi, kā vien saskaņā ar Pārziņa rakstveida norādījumiem, ja vien to darīt nepieprasa tiesību akti.
3. Apstrādātājs veic personas datu apstrādi tikai tajos gadījumos un tādā apjomā, lai izpildītu Līguma saistības (nodrošinātu funkcijas, pakalpojumus, novērstu problēmsituācijas).
4. Apstrādājamie personas datu veidi, datu subjektu un datu apstrādes kategorijas izriet no Līguma, un tās ir:

4.1. Personas datu veidi:

| Personas datu veidi | Datu subjektu kategorijas |
|---|---|
| Vārds, uzvārds | 1.4.2.1.;1.4.2.2. |
| Vecāku vārds, uzvārds | |

| | |
|---|---|
| Adrese | 1.4.2.1. |
| Sakaru līdzekļi | 1.4.2.1;1.4.2.2 |
| Dzimtā valoda | 1.4.2.1. |
| Tautība | 1.4.2.1. |
| Mītnes valsts | 1.4.2.1. |
| Dzimšanas vieta | 1.4.2.1. |
| Dzimšanas datums | 1.4.2.1. |
| Dzimums | 1.4.2.1. |
| Ģimenes stāvoklis | 1.4.2.1. |
| Invaliditāte, īpašās vajadzības | 1.4.2.1. |
| Vēlamais darbs | 1.4.2.1. |
| Vēlamais amats | 1.4.2.1. |
| Darba pieredze | 1.4.2.1. |
| Nodarbinātības vēsture | 1.4.2.1. |
| Izglītības vēsture | 1.4.2.1. |
| Atļaujas (piemēram, vadītāja apliecība) | 1.4.2.1. |
| Licences/sertifikāti | 1.4.2.1. |
| Militārā pieredze | 1.4.2.1. |
| Patenti | 1.4.2.1. |
| Publikācijas | 1.4.2.1. |
| Uzstāšanās vēsture | 1.4.2.1. |
| Kvalifikācija | 1.4.2.1. |
| Kompetences | 1.4.2.1. |
| Vēlamā darba vietas atrašanās | 1.4.2.1. |
| Vēlamais atalgojums | 1.4.2.1. |
| Piedalīšanas organizācijās | 1.4.2.1. |

4.2.

4.2. Datu subjektu kategorijas:

4.2.1. Darba meklētāji;

4.2.2. Pārziņa darbinieki;

4.3. Datu apstrādes veidi:

4.3.1.1. Datu apskate, ko Apstrādātājs veic Pārziņa uzdevumā;

4.3.1.2. Datu pārbaude, ko Apstrādātājs veic Pārziņa uzdevumā;

4.3.1.3. Datu analīze līguma izpildes kvalitātes kontrolei;

4.3.1.4. Datu analīze atskaišu un pārskatu izveidei Pārziņa uzdevumā;

4.3.1.5. Datu nosūtīšana citu informācijas sistēmu pārziņiem un datu saņemšana no citu informācijas sistēmu pārziņiem Sistēmas darbības pārbaudes nolūkos;

4.3.1.6. Datu glabāšana un datu rezerves kopēšana.

## 2. Apstrādātāja pienākumi un tiesības

**2.1.** Apstrādātājs personas datus apstrādā tikai saskaņā ar Līgumu un šī pielikuma 1.punktā noteikto, tai skaitā pēc Pārziņa rakstveida norādījumiem.

**2.2.** Apstrādātājs apliecina un garantē, ka tiks īstenoti atbilstoši tehniskie un organizatoriskie aizsardzības pasākumi tādā veidā, ka Apstrādātāja uz Līguma pamata veiktajā personas datu apstrādē tiks ievērotas Regulas vai Fizisko personu datu aizsardzības likuma (attiecīgi tas normatīvais akts, kurš ir spēkā un ir piemērojams apstrādes brīdī) prasības un tiks nodrošināta datu subjekta tiesību aizsardzība. Par apliecinājumu garantiju izpildei Apstrādātājs attiecībā uz personas datu apstrādi, kuru Apstrādātājs veiks uz Līguma pamata, aizpilda šim pielikumam pievienoto anketu, ko Līguma spēkā esamības laikā aktualizē, tiklīdz ir notikušas izmaiņas attiecībā uz anketā ietverto informāciju vai apliecinājumu daļēji vai kopumā.

**2.3.** Apstrādātājs veic pasākumus, lai nodrošinātu, ka fiziska persona, kas darbojas Apstrādātāja uzdevumā Līguma ietvaros un kam ir piekļuve personas datiem, tos apstrādā atbilstoši Pārziņa norādījumiem. Apstrādātājs nodrošina, ka personas, kuras ir pilnvarotas apstrādāt personas datus Apstrādātāja uzdevumā Līguma ietvaros, ir rakstveidā apņēmušās ievērot konfidencialitāti. Izņēmumi ir pieļaujami, ja Apstrādātājam ir pienākums to veikt saskaņā ar saistošiem tiesību aktiem.

**2.4.** Apstrādātājs īsteno tehniskus un organizatoriskus aizsardzības pasākumus attiecībā uz datu drošību un aizsardzību pret ārēju ielaušanos.

**2.4.1.** Novērtējot atbilstīgo datu aizsardzības līmeni, jo īpaši ņem vērā drošības riskus, tai skaitā nejauši vai prettiesiski nosūtīto, uzglabāto vai citādi apstrādāto personas datu iznīcināšanu, nozaudēšanu, pārveidošanu, prettiesisku piekļuvi tiem vai izpaušanu.

**2.5.** Apstrādātājs ir tiesīgs piesaistīt citus apstrādātājus personas datu apstrādei Līguma saistību izpildei, tikai pēc iepriekšēja rakstiska saskaņojuma ar Pārzini. Šādā gadījumā Līguma 6.3.apakšpunktā noteiktā Apstrādātāja (Izpildītāja) pilnvarotā persona pirms cita apstrādātāja piesaistīšanas informē Līguma 6.2.apakšpunktā noteikto Pārziņa (PASŪTĪTĀJA) pilnvaroto personu par jebkādām iecerētām pārmaiņām saistībā ar papildu apstrādātāju vai apstrādātāja aizstāšanu un sniedz pamatojumu cita apstrādātāja piesaistīšanas nepieciešamībai. Apstrādātājs, piesaistot citu apstrādātāju personas datu apstrādei Līguma saistību izpildes nodrošināšanai Pārziņa vārdā, ir atbildīgs par citam apstrādātājam saskaņā ar regulu noteiktiem personas datu aizsardzības pienākumiem, nodrošinātām tehniskajām un organizatoriskajām prasībām personas datu aizsardzībā, kas noteikti šajā pielikumā un izriet no Līguma.

**2.6.** Apstrādātājs personas datu aizsardzības pārkāpuma gadījumā bez nepamatotas kavēšanās un ne vēlāk kā 48 stundu laikā, kad pārkāpums tam kļuvis zināms, paziņo par pārkāpumu Pārziņa pilnvarotajai personai, kura noteikta Līguma 6.2.apakšpunktā. Pēc datu

aizsardzības pārkāpuma Apstrādātājs sniedz Pārzinim pieprasīto informāciju un sadarbojas, Pārzinim veicot novērtējumu par ietekmi uz datu aizsardzību.

**2.7.** Ne vēlāk kā 5 darba dienu laikā pēc Līgumā noteikto Pušu saistību pilnīgas izpildes Apstrādātājam ir pienākums dzēst visus Līguma izpildes laikā iegūtos un uzglabātos personu datus. Līguma spēkā esamības laikā Apstrādātājs pēc Pārziņa pilnvarotās personas, kura noteikta Līguma 6.2.apakšpunktā, norādījumiem dzēš (iznīcina) personas datus. Apstrādātājam ir pienākums nodrošināt, ka šī pielikuma 2.5.apakšpunktā noteiktajā kārtībā piesaistītie papildu apstrādātāji iegūtos personas datus dzēš 3 darba dienu laikā pēc attiecīgā uzdevuma izpildes.

**2.8.** Apstrādātājs saskaņā ar regulu Pārzinim nodrošina pieejamu visu informāciju, kas nepieciešama, lai apliecinātu, ka tiek pildīti Apstrādātājam paredzētie pienākumi datu aizsardzības jomā, un lai dotu iespēju Pārzinim vai Pārziņa pilnvarotam pārstāvim, iepriekš vienojoties par piedalīšanos, par to pārliecināties.

## 3. Pārziņa tiesības

**3.1.** Pārzinim ir tiesības pieprasīt visu informāciju, kas nepieciešama, lai pārliecinātos, ka tiek pildīti Apstrādātāja pienākumi personas datu aizsardzības jomā..

**3.2.** Saņemt Apstrādātāja veiktā novērtējuma par ietekmi uz datu aizsardzību kopsavilkumu.

## 4. Sadarbība un atbildība

**4.1.** Apstrādātājs sadarbojas ar Pārzini personas datu aizsardzības pārkāpumu (incidentu) izmeklēšanā un novēršanā, regulas un šī pielikuma prasību izpildē, ievērojot Līgumā noteikto pakalpojumu pasūtīšanas un izpildes kārtību.

**4.2.** Apstrādātājs visus datu subjektu un iestāžu pieprasījumus, kas attiecināmi uz personu datiem, kas nonākuši Apstrādātāja rīcībā Līguma izpildes ietvaros, vai atsevišķos gadījumos informāciju par tiem, iesniedz izpildei Pārzinim. Apstrādātājs, ja nepieciešams, sniedz atbalstu Pārzinim attiecīgo pieprasījumu risinājumā, ciktāl tas attiecas uz Līguma ietvaros Apstrādātāja rīcībā esošo informāciju, kas satur personas datus.

## 5. Citi noteikumi

**5.1.** Tiesību aktu grozījumu gadījumā Puses sadarbojas, lai papildinātu un/vai grozītu šo vienošanos, nosakot tajā Pušu saistības atbilstoši spēkā esošajam regulējumam un Līgumā un šajā pielikumā noteiktajai Pušu atbildību sadalījuma būtībai.

**5.2.** Puses neatbild par saistību neizpildi, ja to iemesls ir nepārvaramas varas (*force majour*) apstākļi.

**5.3.** Šis pielikums ir neatņemama Līguma sastāvdaļa. Dokuments ir spēkā visā laika periodā, kamēr Apstrādātājs veic Līgumā noteikto saistību izpildei Pārziņa personu datu apstrādi un laika periodā pēc tās pabeigšanas, kamēr var tikt celti jebkādi prasījumi par saskaņā ar Līguma un šīs vienošanās ietvaros veikto personu datu apstrādi.

**PASŪTĪTĀJS:**

**Nodarbinātības valsts aģentūra**,
Reģ.nr: 90001634668
K. Valdemāra ielā 38 k-1, Rīgā, LV – 1010

/Kristīne Stašāne/

**IZPILDĪTĀJS:**

SIA Uniso
Reģ.nr. 40003562863
Valdlauči 1 - 40, Ķekavas pag., Ķekavas nov.
Konts: LV21HABA0551001217765

/Mārtiņš Rumkovskis/

**5. pielikums**
**pie 2018.gada 15.augusta līguma**
***Par bezdarbnieku uzskaites un reģistrēto vakanču informācijas sistēmas EURES papildinājumu izstrādi***

**Anketa par fizisku personu datu apstrādātāja tehniskajiem un organizatoriskajiem pasākumiem datu aizsardzības nodrošināšanai**

1. Kādi fiziskie aizsardzības līdzekļi ir ieviesti personas datu aizsardzības nodrošināšanai? Vispārīgi aprakstiet tos.

   *Fiziskā apsardze Datu centros,*

   *Piekļuves kontrole (VPN),*

   *Ierobežotu personu saraksts, kam iedota piekļuve,*

   *citi:* _____

2. Kādi tehnoloģiskie aizsardzības līdzekļi ir ieviesti personas datu aizsardzības nodrošināšanai? Vispārīgi aprakstiet tos.

   *Ugunsmūris,*

   *Darbību logošana,*

   *Dalītas pieejas tiesības,*

   *Citi:* _____

3. Kādi organizatoriskie aizsardzības līdzekļi ir ieviesti personas datu aizsardzības nodrošināšanai? Vispārīgi aprakstiet tos.

   *Dalītas definētas pieejas tiesības,*

   *Darbinieku apmācības,*

   *citi:* _____

4. Kā tiek nodrošināta notikuša personas datu aizsardzības pārkāpuma identificēšana - drošības pārkāpums, kura rezultātā notiek nejauša vai nelikumīga nosūtīto, uzglabāto vai citādi apstrādāto personas datu iznīcināšana, nozaudēšana, pārveidošana, neatļauta izpaušana vai piekļuve tiem?

   *Palīdzības dienests problēmu pieteikšanai,*

   *Log failu analīze,*

   *citi:* _____

5. Vai ir noteikta iekšējā kārtība rīcībai personas datu aizsardzības pārkāpuma gadījumā? Vispārīgi raksturojiet to.

   *Noteiktas atbildības,*

   *Noteiktas pārkāpumu ziņošanas un risināšanas procedūras.*

   *citi:* _____

6. Kā tiek nodrošināta spēja laicīgi atjaunot personas datu pieejamību un piekļuvi tiem gadījumā, ja ir noticis fizisks vai tehnisks negadījums?

   *Atbalsta līgumi ar IT sistēmu piegādātājiem / ārējiem uzturētājiem,*

   *Apmācīts personāls,*

   *citi:* _____

7. Kāds process ir ieviests regulārai tehnisko un organizatorisko pasākumu personas datu drošības nodrošināšanai efektivitātes testēšanai, izvērtēšanai un novērtēšanai?

   *Ugunsmūra DNP (LTC),*

   *citi:* _____

8. Vai ir izstrādāta iekšējā kārtība atklāto trūkumu novēršanai? Ja ir, vispārīgi raksturojiet to.

   *citi:* _____

9. Vai un cik bieži un kuru kategoriju darbiniekiem tiek nodrošināta apmācība personas datu aizsardzības jomā?

   *Regulāri 1x 1-2 gados,*

   *citi:* _____

10. Kādi pasākumi tiek veikti, lai nodrošinātu, ka jebkura fiziska persona, kas darbojas apstrādātāja pakļautībā un kam ir piekļuve personas datiem, tos neapstrādā bez apstrādātāja norādījumiem?

    *Iekšējās Darba kārtības noteikumi un instrukcijas par datu apstrādi,*

    *Darba līgums,*

    *citi:* _____

**PASŪTĪTĀJS:**

**Nodarbinātības valsts aģentūra**,
Reģ.nr: 90001634668
K. Valdemāra ielā 38 k-1, Rīgā, LV – 1010

/Kristīne Stašāne/

**IZPILDĪTĀJS:**

SIA Uniso
Reģ.nr. 40003562863
Valdlauči 1 - 40, Ķekavas pag., Ķekavas nov.
Konts: LV21HABA0551001217765

/Mārtiņš Rumkovskis/